1967-09

Determination of optimal compliance and stiffness matrices from experimental data

Villaran Tapia, Cesar F.

Monterey, California. U.S. Naval Postgraduate School

DETERMINATION OF OPTIMAL COMPLIANCE AND STIFFNESS MATRICES FROM EXPERIMENTAL DATA

CESAR F. VILLARAN TAPIA

DETERMINATION OF OPTIMAL COMPLIANCE AND STIFFNESS MATRICES FROM EXPERIMENTAL DATA

by

Cesar F. Villaran Tapia
Lieutenant, Peruvian Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1967

ABSTRACT

A linear structure can be characterized by its compliance matrix C, which is 6x6 symmetrical and positive definite and which relates a force 6-vector $\bar{F}_j$ to a displacement 6-vector $\bar{D}_j$ by the relation $C\bar{F}_j = \bar{D}_j$. The inverse, S, of C, is called the stiffness matrix and satisfies $S\bar{D}_j = \bar{F}_j$. This thesis deals with the problem of finding optimal values of such matrices C and S from experimental determinations of a sufficient number of vector-pairs $(\bar{F}_j, \bar{D}_j)$ which are presumed to contain random errors.

J. E. Brock has introduced this problem area, suggested several different criteria of optimality, and solved some of the corresponding specific problems. This thesis completes the solution to a previously unsolved specific problem of this group and contributes computationally convenient new solutions to another. Moreover, a computer program, originally written for the CDC 1604 has been rewritten, in FORTRAN IV Language, as two programs for the IBM System 360 computer, and the capability has been significantly augmented.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

The writer wishes to gratefully acknowledge the patience and encouragement of his professor and faculty advisor, Dr. John E. Brock, without whose guidance this thesis would not have been followed to its completion.

The writer also wants to thank Mrs. Thomas Miles for her patience and well done job in correcting the English grammar of the original manuscript.

SECTION 1

INTRODUCTION

1.1 Source and Statement of the Physical Problem.

The coordinate system used here is a right handed orthogonal triad, whose unit vectors are denoted by $\bar{e}_1$, $\bar{e}_2$, and $\bar{e}_3$; see figure 1.

A generalized force will be denoted by $\bar{F}$, where

$$\bar{F} = \{ F_1, F_2, F_3, F_4, F_5, F_6 \}$$

is a column matrix. Braces $\{\}$ indicate that a column matrix is intended. The first three elements, F_1, F_2, and F_3, are force components in the $\bar{e}_1$, $\bar{e}_2$, and $\bar{e}_3$ direction respectively, and the last three elements, F_4, F_5, and F_6, are moment components about these axes.

Also let a generalized deflection be denoted by $\bar{D}$, where

$$\bar{D} = \{ D_1, D_2, D_3, D_4, D_5, D_6 \}$$

The first three elements, D_1, D_2, and D_3, are displacements in the $\bar{e}_1$, $\bar{e}_2$, and $\bar{e}_3$ direction respectively, and the last three elements are rotation components about these axes.

It is now presumed that a force $\bar{F}$ is applied to a (suitably constrained) elastic structure, the application being at point P, and the corresponding deflection $\bar{D}$ is observed at P; see figure 1.

It is postulated that a linear relation holds between the elements of $\bar{F}$ and $\bar{D}$, i.e., that there exists a matrix C such that

$$C\bar{F} = \bar{D}$$

It is further postulated that matrix C is symmetric and positive definite, and that it has an inverse matrix S ($S = C^{-1}$) which, likewise, is symmetric and positive definite.

Matrix C is called the compliance (or flexibility) matrix (of the structure) and its elements are called compliance or flexibility coefficients.

Matrix S is called the stiffness matrix, and its elements are called stiffness influence coefficients.

Note that matrices C and S depend not only on the structure, but also on the choice of point P.

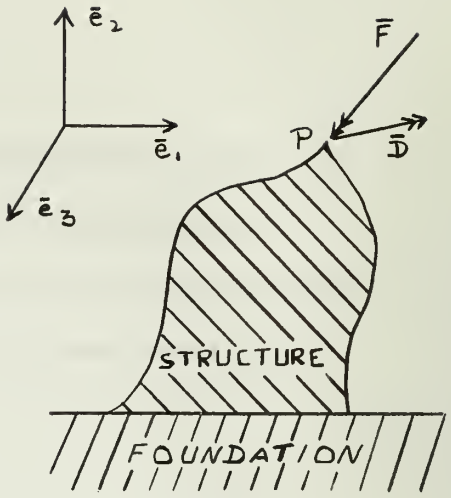

FIGURE 1

There are basically two approaches for obtaining the compliance matrix C, or the stiffness matrix S. One employs elastic theory in an attempt to come up with the elements of C or S. Even with simple structures it may not be easy to write down the elements of these matrices. The theory must be stretched beyond certainty and one obtains only an approximation to the correct result.

The second approach, the use of which leads to this thesis problem, employs experimental data rather than elastic theory to come out with the best C or S. Because of inevitable random errors present in the data, it is impossible to obtain the matric C or S, without using some mathematical procedure that is able to minimize the influence of these errors, and at the same time to assure that matrices C and S are symmetrical and positive definite. Furthermore the data must contain a sufficient number of independent pairs of vectors $\bar{F}$ and $\bar{D}$.

The question of how many pairs of vectors are required is considered in section 7. However in practice, it is anticipated that many more pairs will be available than the theoretical minimum.

1.2 Brief History of this Problem and Related Problems.

Professor John E. Brock [1],[1] of the Mechanical Engineering Department at the Naval Postgraduate School, introduced this problem. He has defined different criteria of optimality and, corresponding to each such criterion, has established a matrix equation, the solution of which will minimize the corresponding errors and permit determining the desired symmetric and positive definite C or S matrix.

In the next chapter there will be given the mathematical descriptions of four criteria of optimality that Professor Brock has introduced in reference [1], and the developments leading to the matrix equations that which must be solved for an unknown square matrix.

A description and an explanation of how to use the computer programs which has been developed for solving these equations will be given later.

The data used to test the programs are not from an actual experiment. An explanation will be given of how sample data, containing random errors, are generated for test purposes.

In this thesis, matrix C has been restricted to be symmetrical and positive definite, as is required in the structural problem under consideration. Two other related situations will be considered in Appendix D.

In general the different criteria of optimality that will be given in the next chapter can be applied to all systems which are described by a matrix equation of the form $C\bar{F} = \bar{D}$, where C, $\bar{F}$, and $\bar{D}$, need only to be comformable for matrix operations.

---

[1] Numbers in brackets refer to references listed in the Bibliography page 43.

SECTION 2

MATHEMATICAL DEVELOPMENT OF THE PROBLEM

2.1 Physical Statement of Criterion One.

It is assumed that there are n pairs of vector observations $\bar{F}$ and $\bar{D}$.

As criterion one, it is desired to determine C so as to minimize the sum of the square of all scalar errors, viz., to minimize

$$\alpha_1 = \sum_{j=1}^{n} \sum_{i=1}^{6} \left(\{ C\bar{F}_j - \bar{D}_j \}_i \right)^2 \qquad (1)$$

where $\{\ \}_i$ = i-th element of vector $\{\ \}$

Now let

$$U = [\ \bar{F}_1, \bar{F}_2, \bar{F}_3, \ldots\ldots\ldots\ldots, \bar{F}_n]$$

$$V = [\ \bar{D}_1, \bar{D}_2, \bar{D}_3, \ldots\ldots\ldots\ldots, \bar{D}_n]$$

These are ($6 \times n$) matrices.

From equation (1)

$$\alpha_1 = \sum_{j=1}^{n} \left[\left(\{ C\bar{F}_j - \bar{D}_j \}_1 \right)^2 + \left(\{ C\bar{F}_j - \bar{D}_j \}_2 \right)^2 + \ldots + \left(\{ C\bar{F}_j - \bar{D}_j \}_6 \right)^2 \right] \qquad (2)$$

Expanding the first term of equation (2)

$$\sum_{j=1}^{n} \left(\{ C\bar{F}_j - \bar{D}_j \}_1 \right)^2 = \left(\{ C\bar{F}_1 - \bar{D}_1 \}_1 \right)^2 + \left(\{ C\bar{F}_2 - \bar{D}_2 \}_1 \right)^2 + \ldots + \left(\{ C\bar{F}_n - \bar{D}_n \}_1 \right)^2 \qquad (3)$$

If all the terms in equation (2), are expanded as in equation (3), it is easy to see that these terms are nothing other than the diagonal terms of the matrix product: $(CU-V)^T(CU-V)$, and therefore it is possible to write

$$\alpha_1 = tr\left[(CU-V)^T(CU-V)\right] \tag{4}$$

where tr stands for the trace of a matrix and superscript T denotes the transposed of a matrix.

2.2 Alternate Criteria.

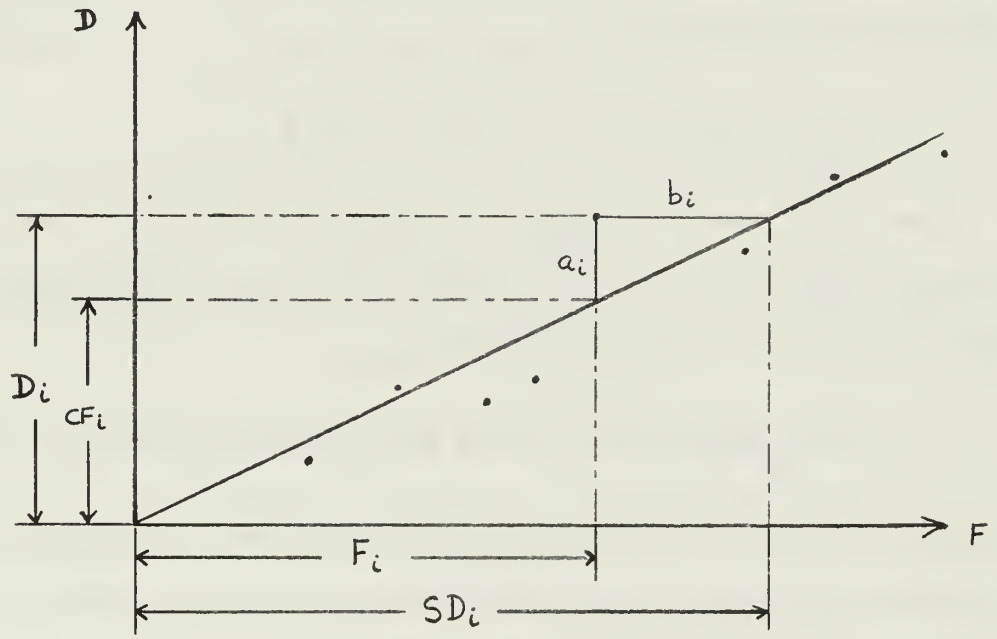

FIGURE 2

As may be seen from Figure 2, in a one dimensional case, criterion one requires minimizing the sum of the square of the vertical errors a_i. It is natural, therefore to consider, as criterion two, minimizing the sum of the squares of the horizontal errors, b_i. In the six dimensional case that this thesis is concerned with, this means minimizing

$$\alpha_2 = \sum_{j=1}^{n} \sum_{i=1}^{6} \left(\{ S\bar{D}_j - \bar{F}_j \}_i \right)^2 \tag{5}$$

Equation (5) is in the same form as equation (1), with the roles of $\bar{F}_j$ and $\bar{D}_j$ interchanged and using the stiffness matrix S instead of its

inverse, the compliance matrix C.

Using the same approach as before this sum can be written in the following way:

$$\alpha_2 = tr\left[(SV-U)^T(SV-U)\right] \tag{6}$$

It is possible too, to minimize the sum of the areas of the "error triangles", whose legs, in a one dimensional case, are a_i and b_i; this is the same as minimizing

$$\alpha_3 = tr\left[(CU-V)^T(U-SV)\right] \tag{7}$$

Now, think also, of minimizing a weighted sum of the square of the horizontal and vertical errors ($\beta_1\alpha_1 + \beta_2\alpha_2$). This criterion number four is the same as minimizing

$$\alpha_4 = tr\left[\beta_1(CU-V)^T(CU-V) + \beta_2(SV-U)^T(SV-U)\right] \tag{8}$$

where β_1 and β_2 are scalar quantities which may be specified so as to designate the relative importance of the two separate parts of which it is composed.

From equation (8), without loss of generality, it is possible to make $\beta_1 = 1$ and $\beta_2 = \beta$, and write equation (8) as

$$\alpha_4 = tr\left[(CU-V)^T(CU-V) + \beta(SV-U)^T(SV-U)\right] \tag{9}$$

Brock [1] also suggested a fifth criterion of optimality but since no progress has been made toward satisfying this criterion, it is not included here.

2.3 Five Theorems on the Trace of a Matrix.

In deriving the matrix equation corresponding to each criterion of

optimality, the following five simple theorems concerning the trace of a matrix will be used. The proofs are simple and are omitted.

THEOREM 1: If the matrix P and the matrix Q are comformable in either order, then each product is square and

$$\text{tr } (PQ) = \text{tr } (QP)$$

THEOREM 2: If the square matrices P and Q may be added, then

$$\text{tr } (P) + \text{tr } (Q) = \text{tr } (P + Q)$$

THEOREM 3: If P is a square matrix, then

$$\text{tr } (P) = \text{tr } (P^T)$$

THEOREM 4: If P is a matrix and x is a scalar, then

$$\left[\frac{\partial P}{\partial x}\right]^T = \frac{\partial P^T}{\partial x}$$

THEOREM 5: If P is a square matrix and x is a scalar, then

$$\frac{\partial tr(P)}{\partial x} = tr\left[\frac{\partial P}{\partial x}\right]$$

Note that theorem 1 permits one to write:

$$\text{tr } (ABCD) = \text{tr } (BCDA) = \text{tr } (CDAB), \text{ and so on.}$$

2.4 Derivation of the Matrix Equations which assure the Optimization.

In sections 2.1 and 2.2 four different criteria of optimality have been given. This section devotes attention to the derivation of a matrix equation corresponding to each criterion. Solving any such equation assures achieving the corresponding optimization.

a) Equation (4) is

$$\alpha_1 = tr\left[(CU-V)^T(CU-V)\right]$$

$$= tr\left(U^TCCU - V^TCU - U^TCV + V^TV\right)$$

Note that instead of C^T, C has been written since C is symmetric. Now let x be any given element of C, and let C* denote $\frac{\partial C}{\partial x}$, then

$$\frac{\partial \alpha_1}{\partial x} = tr\left(U^T C^* C U + U^T C C^* U - V^T C^* U - U^T C^* V\right)$$

$$= tr\left[\left(CUU^T + UU^T C - UV^T - VU^T\right)\left(C^*\right)\right]$$

$$= tr\left[\left(CP + PC - Q\right)\left(C^*\right)\right]$$

where

$$P = UU^T, \text{ and } Q = UV^T + VU^T$$

are obviously symmetric.

Thus, if C satisfies the matrix equation

$$CP + PC = Q \tag{10}$$

then $\frac{\partial \alpha_1}{\partial x} = 0$, for all elements x of C and there is an extremum, which will be minimum, since α_1 is the sum of the squares of real numbers.

Because of the simple relationship between α_1 and α_2, it is evident that a similar equation applies for criterion two.

This equation is

$$SR + RS = Q \tag{11}$$

where Q is as before and $R = VV^T$.

b) Equation (7) is

$$\alpha_3 = tr\left[\left(CU - V\right)^T\left(U - SV\right)\right]$$

The solution will be to determine the matrices C which make α_3 stationary; among these will be one which minimizes α_3.

Knowing that $S = C^{-1}$, and both C and S are symmetric, after some manipulation

$$\alpha_3 = tr\left(U^T C U - V^T U - U^T V + V^T S V\right)$$

As before, let x be any given element of C, and let $C^* = \frac{\partial C}{\partial x}$ and $S^* = \frac{\partial S}{\partial x}$

Since CS = I, then

$$CS^* + C^*S = 0$$

and $$S^* = -SC^*S.$$

Thus

$$\frac{\partial \alpha_3}{\partial x} = tr\left(U^T C^* U - V^T S C^* S V\right)$$

$$= tr\left[\left(UU^T - SVV^T S\right)\left(C^*\right)\right]$$

This will vanish and α_3 will be extremal if S satisfies the matrix equation

$$UU^T = SVV^T S$$

After post and pre-multiplying by C

$$CUU^T C = VV^T \qquad (12)$$

c) Equation (9) is

$$\alpha_4 = tr\left[\left(CU - V\right)^T\left(CU - V\right) + \beta\left(SV - U\right)^T\left(SV - U\right)\right]$$

$$= tr\left[V^T V - U^T C V - V^T C U + U^T C C U + \right.$$

$$\left.\beta\left(V^T S S V - U^T S V - V^T S U + U^T U\right)\right]$$

As before, let x be any given element of C.

Thus, using notation introduced previously, viz. :

$$P = UU^T, \quad R = VV^T, \text{ and } \quad Q = VU^T + UV^T$$

it may be seen that

$$\frac{\partial \alpha_4}{\partial x} = tr\left[-U^T C^* V - V^T C^* U + U^T C^* C U + U^T C C^* U + \right.$$

$$\left. \beta\,(V^T S^* S V + V^T S S^* V - U^T S^* V - V^T S^* U)\right]$$

$$= tr\left[(-VU^T - UV^T + CUU^T + UU^T C)(C^*) + \right.$$

$$\left. \beta\,(SVV^T + VV^T S - VU^T - UV^T)(S^*)\right]$$

$$= tr\left[(-Q + CP + PC)(C^*) + \beta(SR + RS - Q)(-SC^*S)\right]$$

$$= tr\left[\left\{(-Q + PC + CP) - \beta S(SR + RS - Q)S\right\}(C^*)\right]$$

which will vanish if

$$PC + CP - Q = \beta\ S(SR + RS - Q)S \qquad (13)$$

If the right hand side of equation (13) is set equal to zero, the left hand side is precisely the same as equation (10), which corresponds to criterion number one. If the left hand side is set equal to zero, the right hand side is nothing other than equation (11), which corresponds to criterion number two.

SECTION 3

SOLUTION OF EQUATIONS FOR CRITERIA 1 AND 2

Equations (10) and (11) that minimize criteria 1 and 2 respectively are of the same form, where the matrices C, S, P, R, and Q, are all symmetric. The solution by Brock [1], given here, is for equation (10), but is applicable to equation (11), with the roles of P and R interchanged. The solution in equation (10) will be for the unknown matrix C, where in equation (11) is for the unknown matrix S.

In the physical problem of primary interest, matrices C, P, and Q are all 6 x 6, but, for the sake of generality, it will be presumed that they are m x m.

In equation (10)

$$CP + PC = Q$$

there are really only $\bar{N} = \frac{1}{2} m (m + 1)$ unknown elements of C. These unknowns can be arranged in a column matrix according to the following procedure:

Use the notation $M_j = \frac{1}{2} (j - 1) j$.

Given i, j, let $t_{i,j} = M_{max(i,j)} + min (i,j)$.

For example, with i = 3, and j = 5, then

$$t_{3,5} = t_{5,3} = \frac{1}{2} (4)(5) + 3 = 13$$

and

$$t_{2,6} = t_{6,2} = \frac{1}{2} (5)(6) + 2 = 17$$

Table 1 gives values of $t_{i,j}$ for i,j = 1, 2, 3,...,10

Next,a column matrix W of $\bar{N}$ elements is formed from the $\bar{N}$ unknown elements c_{ij} of matrix C. Element c_{ij} of C becomes the $t_{i,j}$th element of W; note that $t_{m,m} = \bar{N}$.

	i=1	2	3	4	5	6	7	8	9	10
j=1	1	2	4	7	11	16	22	29	37	46
2	2	3	5	8	12	17	23	30	38	47
3	4	5	6	9	13	18	24	31	39	48
4	7	8	9	10	14	19	25	32	40	49
5	11	12	13	14	15	20	26	33	41	50
6	16	17	18	19	20	21	27	34	42	51
7	22	23	24	25	26	27	28	35	43	52
8	29	30	31	32	33	34	35	36	44	53
9	37	38	39	40	41	42	43	44	45	54
10	46	47	48	49	50	51	52	53	54	55

TABLE 1. Values of $t_{i,j}$

$$W = \{ w_1, \ldots, w_N \} = \{ c_{11}, \ldots, c_{mm} \}$$

The column matrix Z of $\bar{N}$ elements is formed in a similar way from the known elements $q_{i,j}$ of symmetric matrix Q.

$$Z = \{ z_1, z_2, z_3, \ldots, z_N \}$$

where $z_{t_{i,j}} = q_{ij} / (1 + \delta_{ij})$

and δ_{ij} is the Kronecker's delta.

Finally the coefficient matrix $\bar{P}$ is formed as the sum of m ($\bar{N}$ x $\bar{N}$) matrices P_I, I = 1,2,...,m, each of order $\bar{N}$ x $\bar{N}$, which themselves are formed from the (m x m) matrix $P = UU^T$, according to the following procedure:

P_I is formed from P by inserting rows and columns of zeros, an operation that is denoted by (*)(K)(L) which means insert L rows of zeros after the K^{th} row in the original matrix P, and insert L columns of zeros after the K^{th} column in the original matrix P. The P_I is obtained from P by the operation:

(a) $(*)(0)(M_I)$

(b) $(*)(I)(I-1)$

(c) $(*)(I+1)(I)$

(d) $(*)(I+2)(I+1)$, etc.

until such insertions fall outside the range of the $\bar{N} \times \bar{N}$ size.

Then:

$$\bar{P} = \sum_{I=1}^{m} P_I$$

and solving the system

$$\bar{P}W = Z$$

gives the column matrix W, whose elements can then be rearranged to produce the desired matrix C.

Bickley and McNamee [4] have considered a more general equation of the form:

$$AZ + ZB = F$$

and they give four methods of solution, which they have named: The Big Matrix, the Irrational Solution, the Semi-Rational Solution, and A Rational Solution.

The method given by Larsen [5], and the method given by Brock [1] and reproduced here, appear to be additional distinct methods; both, however, probably are closely related to what Bickley and McNamee refer to as the "big matrix", in which A = B.

As the results obtained with the method given here is quite satisfactory, no further study has been done with the methods given in reference [4], but it might be of interest to one working with this problem to try suitably specializing Bickley and McNamee's methods.

SECTION 4

SOLUTION OF EQUATION FOR CRITERION 3

To solve equation (12), postmultiply by UU^T; then

$$CUU^TCUU^T = VV^TUU^T \tag{14}$$

$$CUU^T = (VV^TUU^T)^{\frac{1}{2}} \tag{15}$$

In equation (14), UU^T and VV^T, both are symmetric, but the product VV^TUU^T is non-symmetric.

Brock [6], has solved equation (15), using a highly sophisticated computer library subroutine, based on a Laguerre-Hessenberg algorithm, to find eigenvalues of VV^TUU^T, which are then used in a method described by Frazer, Duncan, and Collar [3], in order to find $CUU^T = (VV^TUU^T)^{\frac{1}{2}}$, from which C may easily be obtained. This procedure is available in a computer program for the FORTRAN 63 language to be used in the CDC 1604 computer.

Owing to the difficulties in converting this program to FORTRAN IV language used in the IBM System 360 computer, recently installed at the Naval Postgraduate School, the following two other procedures were developed and have successfully been applied to solve equation (12).

a) Iterative procedure for finding the square root of a non-symmetric matrix.

Pulay [7], has given a method that determines the principal square root of a positive definite symmetric matrix using an iteration procedure.

Using his idea with some modification, a method for obtaining the square root of a positive definite non-symmetric matrix has been successfully used and applied to several examples.

In equation (15)

$$CUU^T = (VV^TUU^T)^{\frac{1}{2}}$$

let $VV^TUU^T = A$

and $A^{\frac{1}{2}} = D + B$ (16)

where D is a diagonal matrix, whose elements are the positive square roots of the diagonal elements of A, and B is a matrix formed from $\tilde{C}UU^T - D$, and $\tilde{C}$ is an approximation of the matrix C that will minimize the criterion number 3.

The matrix $\tilde{C}$ can be obtained from the equation

$$\tilde{C}U = V$$

$$\tilde{C}UU^T = VU^T$$

and

$$\tilde{C} = VU^T(UU^T)^{-1}$$

This is one way of finding the matrix $\tilde{C}$, suggested by analogy to Pulay's method. However, in the computer program used in this thesis to solve equation (15), the matrix $\tilde{C}$ is taken to be the matrix C that satisfies criterion number 2. This is done in order to save computer time.

From equation (16)

$$(D + B)^2 = A$$

$$DB + BD = A - D^2 - B^2$$

This equation can not be solved for matrix B due to the term B^2, but following Pulay's procedure it can be transformed into a simple iteration formula

$$DB_k + B_kD = A - D^2 - B_{k-1}^2$$

and the (i,j)-th element of B_k, can be obtained in a simple manner from the formula

$$(B_k)_{ij} = \frac{1}{D_i + D_j}\left[(A - D^2)_{ij} - (B_{k-1}^2)_{ij}\right]$$

To obtain $(B_k)_{ij}$, the iteration procedure must be carried out until the difference between two successive approximations of $(B_k)_{ij}$ become less than a specified allowable error, i.e.

$$(B_k)_{ij} - (B_{k-1})_{ij} = \Delta$$

where Δ in the actual program is 10^{-10}, and the $()_{ij}$ implies comparing element by element.

After matrix B has been obtained

$$C = (D + B)(UU^T)^{-1}$$

or

$$C = (VV^T UU^T)^{\frac{1}{2}}(UU^T)^{-1}$$

The procedure described above leads to output 3A in the actual computer program. Output 3B is the solution of basically the same equation, but instead of equation (12) being postmultiplied by UU^T, it is premultiplied resulting in

$$UU^T C = (UU^T VV^T)^{\frac{1}{2}}$$

and using the same procedure as before the square root of $UU^T VV^T$ can be obtained, and therefore the unknown matric C.

Output 3C is simply the arithmetic average of 3A and 3B.

b) The second procedure that solves equation (12) uses some matrix algebra manipulation, and instead of working with a non-symmetric matrix , it requires taking the square roots of two different symmetric matrices. This procedure was called to the attention of the writer by Professor Brock.

Observe that starting with equation (12), $CUU^T C = VV^T$ and premultiplying by U^T and postmultiplying by U, the following matrix equation is obtained

$$U^T CUU^T CU = U^T VV^T U$$

and

$$U^TCU = (U^TVV^TU)^{\frac{1}{2}}$$

where U^TVV^TU is a symmetric matrix, but as U and V have dimensions (6xn), where n is the number of pair of vectors available, this formulation of the problem requires taking the square root to an (nxn) symmetrix matrix, with the disadvantage of dealing with a big matrix, for example (1000x1000).

Accordingly, return to equation (12)

$$CUU^TC = VV^T$$

Let $UU^T = A$, and without much loss of generality assume that the eigenvalues of A are distinct. In this problem A is a (6x6) matrix.

Then

$$A = ML^2M^T$$

where M is the modal matrix of A

and L^2 is the spectral matrix of A.

This is true since A is symmetrical, L^2 is simply a diagonal matrix of the eigenvalues of A, and M is a square matrix whose columns are the corresponding eigenvectors of A. M is orthogonal so that $M^T = M^{-1}$.

Since L^2 is a diagonal matrix, it is easy to find its square root, L. Theoretically there are a number of different choices of L since either the positive or the negative square root of each eigenvalues can be chosen, but in this application it is convenient and satisfactory to use only the positive values.

Now

$$\begin{aligned} A = ML^2M^T &= MLLM^T \\ &= MLL^TM^T \\ &= EE^T \end{aligned}$$

where $E = ML$ is (mxm), i.e., 6 x 6 in the applied problem.

Let $B = VV^T$

and equation (12) now takes the form

$$CEE^TC = B$$

$$E^TCEE^TCE = E^TBE = D$$

where D is an (m×m) symmetric matrix. Thus,

$$E^TCE = D^{\frac{1}{2}}$$

Like matrix A, D can be written in terms of its modal and spectral matrices

$$D = m1^2m^T$$

$$D^{\frac{1}{2}} = m1m^T$$

Therefore

$$E^TCE = m1m^T$$

Now $$E^{-1} = L^{-1}M^{-1} = L^{-1}M^T$$

where L^{-1} is simply the diagonal matrix of the reciprocals of the corresponding elements of L,

Thus

$$C = ML^{-1}\, m1m^TL^{-1}M^T$$

This procedure has been written in the FORTRAN IV language to be used in the IBM System 360 computer, and in the actual program is output 3D.

SECTION 5

SOLUTION OF EQUATION FOR CRITERION 4

The equation (13), that minimize the criterion 4, is nonlinear in the unknown matrix C ($= S^{-1}$), and an iteration method is used for its solution.

Given β , choose a convenient $\rho \leqq \beta$. Now from equation (13), consider the iterative scheme indicated by

$$C_i P + P C_i - Q = \rho S_i (S_i R + R S_i - Q) S_i \qquad (17)$$

where

$$S_i = C_{i-1}^{-1}$$

and $$C_o = \frac{\bar{C}_1 + \rho \bar{C}_2}{1 + \rho}$$

$\bar{C}_1$ is the matrix that satisfies criterion 1

$\bar{C}_2$ is the matrix that satisfies criterion 2

Now, having C_o, S_1 is determined, from which the right hand side of equation (17) can be evaluated, and the following equation can be obtained:

$$C_i P + P C_i = Q^* \qquad (18)$$

where

$$Q^* = Q + \rho S_i (S_i R + R S_i - Q) S_i$$

Equation (18) is of the same form as equation (10), and can be solved using the same procedure.

The C_i obtained from equation (18), is used again to obtain S_{i+1}, and a new C_{i+1} is obtained using the iterative procedure until two successive approximations of C_i become less than a specified allowable error, viz.

$$(C_{i+1})_{jk} - (C_i)_{jk} = \Delta$$

where Δ in the actual program is taken as 10^{-5}, and $()_{jk}$ implies comparing element by element.

When the convergence is achieved, ρ is increased and the iterative procedure is carried out again. This procedure is followed until finally convergence is obtained with $\rho = \beta$.

The reason for using values of ρ initially less than β is to assure and optimize convergence of the entire process. For example, in one study where $\beta = 0.8$, it was found convenient to use the sequence

$$\rho = 0.1, 0.2, 0.3, \ldots, 0.8.$$

SECTION 6

DIGITAL COMPUTER PROGRAMS

Two main computer programs have been written in FORTRAN IV Language to be used with the IBM System 360 computer. These programs are called CECI 1 and CECI 2. They make use of several subroutines and to assure accuracy the main programs and the subroutines employ double precision arithmetic.

In program CECI 1, the following subroutines are used:

ADDEM	AINVM
BIGCM	BIGSM
BLANCA	JACVAT
MAUSS3	OUTPU
PRINM	RECIP
REDEM	SQRERT
SQROOT	SQUAM
TRACM	TRANM
ULTMA	ULTMB

In the program CECI 2, the following subroutines are used:

ADDEM	AINVM
BIGCM	BIGSM
BLANCA	MAUSS3
OUTPUT	PRINM
REDEM	SQUAM
TRACM	TRANM
ULTMA	ULTMB

Both programs make use of a number of functions from the monitor library tape which are not listed here.

Program CECI 1 solves the equations that satisfy criteria numbers 1, 2, and 3. The program gives one output for each of criteria 1 and 2, and four outputs for criterion number 3. For each of these outputs the program prints the compliance matrix C, and for each such matrix, computes and prints out three criteria numbers, as follows:

CRITERION NUMBER ONE $= tr\,[(CU - V)^T(CU - V)]$

CRITERION NUMBER TWO $= tr\,[(SV - U)^T(SV - V)]$

CRITERION NUMBER THREE $= tr\,[(CU - V)^T(U - SV)]$

Program CECI 2 solves the equation that satisfies criterion number 4, prints the compliance matrix C, and gives four criterion numbers; the first three are the same as in CECI 1 and the fourth is given by:

CRITERION NUMBER FOUR $= tr\,[(CU - V)^T(CU - V) + \beta\,(SV-U)^T (SV- U)]$

Most of the matrices are regarded as doubly subscripted arrays as follows:

If there are M rows and N columns, the array in the computer memory has dimension (M + 1, N); for example: if the matrix has 6 rows and 50 columns, the dimension of the array is (7 x 50). This extra row permits the identification of the matrix for debugging purposes and gives this information to those subroutines which use it for processing or monitoring purposes.

Let $e_{i,j}$ refer to the array and let $a_{i,j}$ refer to the matrix. Then $e_{1,1}$ is an integer identifying the array and the matrix. $e_{1,2}$ and $e_{1,3}$ are integers M and N respectively. For the remainder of the elements, $e_{i+1,j} = a_{i,j}$. The other matrices used in the main programs that have the dimensions (N,N), are used as scratch matrices. They are needed when calling some subroutines that have dummy subscripted variables, due

to the mode of storage (column-wise) used by the IBM System 360 computer. The array for an M-element column matrix has dimension (M + 1), where the first element is the integer M.

The programs can be used in two different ways: (1) for classroom or more investigation purposes and (2) for use with data from an actual experiment. This is done by means of the card called MODE, where MODE = 1, 2, respectively.

If MODE = 1, the program will read the matrices C and U from the data cards, and generate matrix V = CU. For testing purposes random errors are introduced in matrices U and V. The explanation of how these errors are introduced will be given in the next section.

If MODE = 2, the programs will read the matrices U and V from the data cards. This data would be obtained from an actual experiment.

The explanation of how to use these programs, and how the data is punched in cards will be given in Appendix A.

Unless other FORMAT statements are called for, the only printed output is as follows:[2]

a) In main program CECI 1, with MODE = 1.

(1) MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

Listing of matrix P in FORMAT (1X, 8D12.4)

(2) MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS, NN COLUMNS

Listing of matrix BE in FORMAT (1X, 8D12.4)

---

[2] The nomenclature used in the outputs of the programs CECI 1 and CECI 2 is given in Appendix C; see pages 76 and 89.

(3) MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS, NN COLUMNS

Listing of Matrix EB in FORMAT (1X, 8D12.4)

(4) ERROR IN BLANCA = (number given)

(5) MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS, NN COLUMNS

Listing of matrix U in FORMAT (1X, 8D12.4)

(6) MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS, NN COLUMNS

Listing of matrix V in FORMAT (1X, 8D12.4)

(7) ELAPSED TIME = (time given in sec.)

(8) OUTPUT NO. 1, MINIMIZES V SQUARE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 17, 6 ROWS, 6 COLUMNS

Listing of desired compliance matrix in FORMAT (1X, 8D12.4)

CRITERION NUMBER 1 EQUALS (value given)

CRITERION NUMBER 2 EQUALS (value given)

CRITERION NUMBER 3 EQUALS (value given)

(9) ELASPED TIME = (time given in sec.)

(i0) OUTPUT NO. 2, MINIMIZES U SQUARE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 33, 6 ROWS, 6 COLUMNS

Listing of desired compliance matrix in FORMAT (1X, 8D12.4)

CRITERION NUMBER 1 EQUALS (value given)

CRITERION NUMBER 2 EQUALS (value given)

CRITERION NUMBER 3 EQUALS (value given)

(11) ELAPSED TIME = (time given in sec.)

(12) OUTPUT NO. 3A, MINIMIZES TRIANGLE ERROR

DESIRED MATRIX IS

MATRIX NUMBER XXX, 6 ROWS, 6 COLUMNS

(XXX may have any large integer value, due to the iteration procedure used to obtain output 3A)

Listing of desired compliance matrix in FORMAT (1X, 8D12.4)

CRITERION NUMBER 1 EQUALS (value given)

CRITERION NUMBER 2 EQUALS (value given)

CRITERION NUMBER 3 EQUALS (value given)

(13) ELASPED TIME = (time given in sec.)

(14) OUTPUT NO. 3B, MINIMIZES TRIANGLE ERROR

Same as number (12)

(15) ELAPSED TIME = (time given in sec.)

(16) OUTPUT NO. 3C, MINIMIZES TRIANGLE ERROR

Same as number (12)

(17) OUTPUT NO. 3D, MINIMIZES TRIANGLE ERROR

Same as number (12)

(18) ELAPSED TIME = (time given in sec.)

(19) TOTAL ELAPSED TIME = (time given in sec.)

b) In main program CECI 1, with MODE = 2.

(1) MATRIX U FOLLOWS

MATRIX NUMBER 1, 6 ROWS, NN COLUMNS

Listing of matrix U in FORMAT (1X, 8D12.4)

(2) MATRIX V FOLLOWS

MATRIX NUMBER 2, 6 ROWS, NN COLUMNS

Listing of matrix V in FORMAT (1X, 8D12.4)

(3) ELAPSED TIME = (time given in sec.)

(4) OUTPUT NO. 1, MINIMIZES V SQUARE ERROR
DESIRED MATRIX IS

MATRIX NUMBER 14, 6 ROWS, 6 COLUMNS

Listing of desired compliance matrix in FORMAT (1X, 8D12.4)

CRITERION NUMBER 1 EQUALS (value given)

CRITERION NUMBER 2 EQUALS (value given)

CRITERION NUMBER 3 EQUALS (value given)

From here the printed out is the same as when MODE = 1.

c) In main program CECI 2, with MODE = 1.

(1) MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

Listing of matrix P in FORMAT (1X, 8D12.4)

(2) MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS, NN COLUMNS

Listing of matrix BE in FORMAT (1X, 8D12.4)

(3) MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS, NN COLUMNS

Listing of matrix EB in FORMAT (1X, 8D12.4)

(4) ERROR IN BLANCA = (value given)

(5) MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS, NN COLUMNS

Listing of matrix U in FORMAT (1X, 8D12.4)

(6) MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS, NN COLUMNS

Listing of matrix V in FORMAT (1X, 8D12.4)

(7) ELAPSED TIME = (time given in sec.)

(8) ELAPSED TIME = (time given in sec.)

(9) NUMBER OF ITERATIONS = (number given)

(10) OUTPUT NO. 4, MINIMIZES V + BETA x U SQUARE ERRORS

DESIRED MATRIX IS

MATRIX NUMBER XXX, 6 ROWS, 6 COLUMNS

Listing of desired compliance matrix in FORMAT (1X, 8D12.4)

BETA = (number given)

CRITERION NUMBER 1 EQUALS (value given)

CRITERION NUMBER 2 EQUALS (value given)

CRITERION NUMBER 3 EQAULS (value given)

CRITERION NUMBER 4 EQUALS (value given)

(11) ELAPSED TIME = (time given in sec.)

(12) TOTAL ELAPSED TIME = (time given in sec.)

d) In main program CECI 2, with MODE = 2.

(1) MATRIX U FOLLOWS

MATRIX NUMBER 1, 6 ROWS, NN COLUMNS

Listing of matrix U in FORMAT (1X, 8D12.4)

(2) MATRIX V FOLLOWS

MATRIX NUMBER 2, 6 ROWS, NN COLUMNS

Listing of matrix V in FORMAT (1X, 8D12.4)

From here the printed out is the same as when MODE = 1.

Actual printouts for both programs, when MODE = 1, are shown in Appendix C.

The listing of the main programs and subroutines will be given in Appendices A and B respectively.

SECTION 7

TESTING OF THEORY AND DIGITAL COMPUTER PROGRAMS

In this investigation the digital computer programs written to solve the matrix equations that satisfy the four criteria of optimality have not been tested using actual (mechanical) experimental data. The data have been generated by assuming a symmetric positive definite matrix C, and a set of vector forces $\{\bar{F}_j\}$ in the form of matrix U. From these two matrices the corresponding set of vector deflections $\{\bar{D}_j\}$ in the form of matrix V were obtained as V = CU.

To simulate experimental data, matrices U* and V* were formed in the following way:[3]

$$U^* = U + E(U)$$

$$V^* = V + E(V)$$

where E(U) and E(V) are matrices of errors.

Then matrices U* and V* were employed as if they were composed of experimental data, to be used in the programs to recover an approximation matrix C* for C.

To form error matrices E(U), and E(V), a mathematical procedure was devised to perform some alterations in each element of the matrices, U and V, in a random manner with an upper limit to the absolute value of the percentage of error being introduced. This procedure has not been tested for randomness, but the method used to obtain the alterations suggests that the values of the elements in matrices E(U) and E(V) are indeed pretty good random errors.

---

[3]Note that the use of the asterisk,*, is different in this section from its use in section 2.

The random errors are generated by a subroutine called BLANCA. Because the first row of most of the matrices are used for identifying purposes, the subroutine makes use of the element(1,1), which is an identifying integer called NUMBER.

Subroutine BLANCA modifies a given matrix $A = [\ a_{ij}\ ]$ by introducing random errors $E = [e_{ij}]$, resulting in a modified matrix $A^* = A+E$. One specifies the maximum percentage error to be applied.

First a number $\rho = 3.6327 +$ (matrix identification NUMBER) is formed. ρ is increased for each element by adding a real constant (3.6327). It is not allowed to be greater than 9.999; this is done by subtracting a real constant (8.4153) each time ρ becomes greater than 9.999.

Next an exponent γ = (I/10+J/25+0.62832) is formed. Next the number $\delta = \rho^{\gamma}$ is formed. Next the number η = (fractional part of δ) is obtained. Finally the relative error is determined as η x r, where r is the preselected maximum relative error. Then

$$e_{ij} = \pm\ \eta \times r \times a_{ij}$$

The algebraic sign is given by the first digit in the decimal expression of η. If this is even, the sign is positive; if this is odd, the sign is negative.

A special treatment is used if $a_{ij} = 0$, for which $e_{ij} = \pm\ \eta$, the sign being determined as before. This has obvious disadvantages if the average value of the elements in A is small, and it is suggested that future work make an appropriate modification to the present procedure.

In retrospect it appears that perhaps a more truly random variation could have been attained by using η = (fractional part of 100 δ), say. Furthermore, it would have been desirable to incorporate such randomizing influences as the date, or local time, or job sequence which

should become available internally to the computer, but which were not available at the time this work was done.

For convenience in comparing values of C* with C, to see how good the recovered matrix C* was, most of the compliance matrices C used were diagonal matrices. However, there is no loss in generality as may be seen from the following argument.

Starting with the matrix equation CU = V, with diagonal C, and premultiplying this equation by an orthogonal matrix M, it can be written as

$$MCM^{T}MU = MV \tag{19}$$

where the insertion of $M^{T}M$ in equation (19) between C and U does not alter the equality, due to the fact that $M^{T}M = I$.

Now let

$$MCM^{T} = \bar{C}$$

$$MU = \bar{U}$$

$$MV = \bar{V}$$

equation (19) can be written now as

$$\bar{C}\bar{U} = \bar{V} \tag{20}$$

Equation (20) has the same form as original matrix equation CU = V, but with the difference that starting with C (a diagonal matrix), the new symmetrical matrix $\bar{C}$ is no longer a diagonal matrix, and instead all its elements are generally non-zero.

Thus a general problem $\bar{C}\bar{U} = \bar{V}$ could be written in the particular form CU = V, with diagonal C, if only the orthogonal matrix M, which diagonalizes C were known. $C = M^{T}\bar{C}M$, $U = M^{T}\bar{U}$, and $V = M^{T}\bar{V}$.

The particular validity of this argument was verified by two types of numerical experiment. In the first type, nondiagonal C's were used, errors were introduced in U and V = CU, and the recovered matrix C* was

successfully compared with the original C. In the second type, an orthogonal matrix M was generated, and matrices $\bar{C}$, $\bar{U}$, and $\bar{V}$ were formed using M, diagonal C, U, and V. Then random errors were introduced in both sets of data (U, V and $\bar{U}$,$\bar{V}$) and C* and $\bar{C}$* were obtained. Comparison of $M^T C^* M$ with C was quite satisfactory.

From theory the minimum number of independent pairs of vectors $\bar{F}$ and $\bar{D}$ needed to solve the problem is 4. This is because in the compliance matrix C there are 21 unknowns, and 21 independent equations must be formed. Thus would require four pairs of 6-vectors $\bar{F}$ and $\bar{D}$ since the notion of 3½ pairs is excluded. However, the particular methods employed in the programs seem to demand a larger number. The minimum number of pairs of vectors needed to obtain "good" results was six.

In one case, a problem with 24 pairs of vectors was solved twenty times, using a maximum random error of ten percent. The errors were introduced in a different manner for each of the solutions. From the twenty compliance matrices obtained in this way, an average compliance matrix was obtained. This matrix looks "very good", when compared element by element with the original compliance matrix. This indicates that if the value of n is so large as to overload computer memory or as to prove inconvenient for loading data, the U and V matrices may be conveniently partitioned, a number of "smaller problems" dealt with, and the answers averaged. One could regard the case described as forming a single problem with n = 480 which was actually treated as the average of the results of twenty problems, each having n = 24.

SECTION 8

CONCLUSIONS AND RECOMMENDATIONS

There does not seem to be any direct way of testing the validity of the theory and the computer programs which implement it other than by taking actual examples. These examples were constructed by assuming values for matrices C and U and generating matrix V = CU. Then, in most cases, errors were applied so as to get modified matrices U* and V*. Then, from these matrices, considered as input data to the program, the various procedures were used to recover an optimal value, C*, which is then compared with the original C. No useful or convenient method has been found for assessing the accuracy of the recovery other than by comparing term by term, and, to simplify this task, in most cases C was assumed to be a diagonal matrix. It was shown in Section 7 that there is no loss in generality in doing this.

One of the things that might be expected to influence the ease and accuracy of the recovery is the relation between the eigenvectors of the matrix C. Tests were conducted, at the one extreme, with diagonal C having identical diagonal elements, and at the other extreme, with diagonal matrices having diagonal elements differing by a factor of 3125. In all, fairly extensive numerical work was investigated and various maximum percentage errors were introduced. The output produced as a result of these studies is much too volumnious to be reported in detail and it has not been found possible to reduce the volume of information in a satisfactory quantitative way. Accordingly, it is necessary to restrict remarks to those of a general nature which will be made in the following. In general, it may be concluded that the theory and the program have been proved to be accurate and effective, and in combination they provide a useful tool for analysing the data from certain types of

structural and mechanical experiments.

The following matrices have been used as compliance matrices C. from which the matrices C* were recovered.

(1)
$$\begin{bmatrix} 9. & 0 & 0 & 0 & 0 & 0 \\ 0 & 18. & 0 & 0 & 0 & 0 \\ 0 & 0 & 5. & 0 & 0 & 0 \\ 0 & 0 & 0 & 12. & 0 & 0 \\ 0 & 0 & 0 & 0 & 8. & 0 \\ 0 & 0 & 0 & 0 & 0 & 27. \end{bmatrix}$$

(2)
$$\begin{bmatrix} 1. & 0 & 0 & 0 & 0 & 0 \\ 0 & 2. & 0 & 0 & 0 & 0 \\ 0 & 0 & 3. & 0 & 0 & 0 \\ 0 & 0 & 0 & 4. & 0 & 0 \\ 0 & 0 & 0 & 0 & 5. & 0 \\ 0 & 0 & 0 & 0 & 0 & 6. \end{bmatrix}$$

(3)
$$\begin{bmatrix} 1. & 0 & 0 & 0 & 0 & 0 \\ 0 & 10. & 0 & 0 & 0 & 0 \\ 0 & 0 & 100. & 0 & 0 & 0 \\ 0 & 0 & 0 & 100. & 0 & 0 \\ 0 & 0 & 0 & 0 & 10. & 0 \\ 0 & 0 & 0 & 0 & 0 & 1. \end{bmatrix}$$

(4)
$$\begin{bmatrix} 7. & 0 & 0 & 0 & 0 & 0 \\ 0 & 7. & 0 & 0 & 0 & 0 \\ 0 & 0 & 7. & 0 & 0 & 0 \\ 0 & 0 & 0 & 7. & 0 & 0 \\ 0 & 0 & 0 & 0 & 7. & 0 \\ 0 & 0 & 0 & 0 & 0 & 7. \end{bmatrix}$$

(5)
$$\begin{bmatrix} 5. & 0 & 1. & 0 & 0 & 1. \\ 0 & 5. & 0 & 0 & 2. & 0 \\ 1. & 0 & 5. & 0 & 0 & 1. \\ 0 & 0 & 0 & 5. & 0 & 0 \\ 0 & 2. & 0 & 0 & 5. & 0 \\ 1. & 0 & 1. & 0 & 0 & 5. \end{bmatrix}$$

(6)
$$\begin{bmatrix} 1. & 0 & 0 & 0 & 0 & 1. \\ 0 & 2. & 0 & 0 & 1. & 0 \\ 0 & 0 & 3. & 1. & 0 & 0 \\ 0 & 0 & 1. & 4. & 1. & 0 \\ 0 & 1. & 0 & 1. & 5. & 1. \\ 1. & 0 & 0 & 0 & 1. & 6. \end{bmatrix}$$

(7)
$$\begin{bmatrix} 1. & 0 & 0 & 0 & 0 & 0 \\ 0 & 5. & 0 & 0 & 0 & 0 \\ 0 & 0 & 25. & 0 & 0 & 0 \\ 0 & 0 & 0 & 125. & 0 & 0 \\ 0 & 0 & 0 & 0 & 625. & 0 \\ 0 & 0 & 0 & 0 & 0 & 3125. \end{bmatrix}$$

OUTPUT No. 1 Gives the desired compliance matrix that satisfies criterion one. With this procedure the types of matrices numbered 1, 4, 5, and 6, have been recovered quite successfully using up to 35% maximum random

errors. Matrices numbered 3 and 7 were recovered with up to 5% maximum random errors, and matrix 2 was recovered with errors up to 25%. (Where no errors were introduced, each original was accurately recovered).

OUTPUT No. 2 — Gives the desired compliance matrix that satisfies criterion two. All the matrices were recovered same as the originals when no errors were involved. Matrices numbered 1, 2, 3, 4, and 5 have been recovered quite successfully using up to 35% random errors, and matrices numbered 6 and 7 with up to 25% random errors.

OUTPUT Nos. 3A,3B, 3C — Gives the desired compliance matrix that satisfies criterion three, using the procedure that obtains the square root of a non-symmetric matrix. All the matrices were recovered same as the originals when no errors were involved. With matrices numbered 1 to 6 this procedure has been worked successfully with up to 35% random errors, and with matrix numbered 7 with up to 20% random error.

OUTPUT No. 3D — Gives the desired compliance matrix that satisfies criterion three, using the procedure that takes square roots of two different symmetric matrices. Matrix numbered 1 is the only one that has not been recovered same as the original when no errors were involved. In the light of other results obtained, this failure to recover matrix 1 to a very high degree of accuracy is unexpected and it has not been satisfactorily

explained. Matrices numbered 1 to 6 using up to 35% random error, have been recovered quite successfully, and matrix numbered 7 with up to 25%.

OUTPUT No. 4 Gives the desired compliance matrix that satisfies criterion four. Using any value for BETA, when no errors were involved, all the matrices were recovered same as the originals. When the error was increased, better results were obtained for small BETA's. The same remarks given for OUTPUT No. 1, applies to this output.

To have a better idea how these results look, see Appendix C, where the computer solutions for a particular problem with zero and ten percent random errors, are given.

The work presented here is just the beginning and not the last word in this new area; more work and investigation should be done. Some suggestions are given below:

1) Solution of equations that satisfy criteria 1, 2, and 4, can be investigated using the procedures given by Bickley and McNamee [4], or possible modifications to take advantage of the fact that matrices A and B are equal.

2) Investigate how large maximum random error percentages one can give to matrices U and V. Give to these matrices different maximum percentages of errors, e.g., to matrix U, 5%, and to matrix V, 15%.

3) There exists a fifth criterion given by Brock [1], that has not been investigated yet.

4) Devise a new iterative procedure for the equation that satisfies criterion four.

5) Devise a convenient criterion which measures the "overall" accuracy of recovery of a matrix C. At present, one merely compares element by element.

6) Employ advanced and sophisticated statistical theory for the analysis of errors and the accuracy of recovery. This would seem to imply constructing an adequate set of definition and the use of probability spaces, etc. The present analysis, in which general conclusions are attempted from specific cases, leaves much to be desired.

As things stand now, the engineering analyst has, in this theory and program implementation, a four edged tool which can be used for the analysis of experimental data. The decision of which criterion yields the "best" result is one which must be made on the basis of experience with the physical problem.

BIBLIOGRAPHY

1. Brock, John E. Second Progress Report, contract N161-26327 with John F. Wax, Inc., and United States Naval Academy, May 15, 1966.

2. Martin, Harold C. Introduction to Matrix Methods of Structural Analysis, McGraw-Hill, 1966.

3. Frazer, R. A., Duncan, W. J., and Collar, A. R. Elementary Matrices, Cambridge University Press and the Macmillan Co., Cambridge and New York, 1947.

4. Bickley, W. G., and McNamee, J. Matrix and other direct methods for the solution of systems of linear difference equations, Philosophical Trans. of the Royal Society of London, Series A., No. 1005, v. 252, 21 January 1960: 69-131.

5. Larson, H. J. Least squares estimation of the components of a symmetrical matrix. Techonometrics, v. 8, No. 2, May 1966: 360-362.

6. Brock, John E. Third Progress Report, contract N161-26327 with John F. Wax, Inc., and United States Naval Academy, July 15, 1966.

7. Pulay, Peter. An Iterative Method for the Determination of the Square Root of a Positive Definite Matrix. Zeit. d.Ang.Math. u. Mech. (ZAMM), v. 46, 1966: 151.

APPENDIX A

INSTRUCTIONS FOR USE AND LISTING OF PROGRAMS

A-1 Preparation of Data Deck and Final Assembly of Programs.

The preparation of the data deck is the same for programs CECI 1 and CECI 2, but has some alterations depending on whether MODE = 1 or MODE = 2.

In what follows the instructions of how to prepare the data deck is given for both types of modes.

The first card is bbbbbbb6NNbbb1bb....., where b = blank, the 6 is the number of rows, NN is the number of columns, and 1 means first matrix to be read. When MODE = 1, NN = 06, is the number of columns in matrix C; when MODE = 2, NN varies from 06 to 50, and is the number of columns in matrix U. (Thus U is limited to a maximum of 50 columns).

Following this card, comes the elements of matrix C or U as the case may be, punched in five entries to a card, in FORMAT (5F14.6). Entries for the first row are thus entered until all elements for the row have been punched. Then, starting with a new card, entries for the second row are punched, and so on, until the entries for the sixth row have been entered. Each row starts with a new card.

Card bbbbbbb6NNbbb2bb....., follows. Then the entries for the matrix U (if MODE = 1) or matrix V (if MODE = 2) are punched in the same manner as before.

Example: Punch cards as in Figure A-1, for the following 6x6 diagonal matrix as C matrix.

$$C = \begin{bmatrix} 1. & 0 & 0 & 0 & 0 & 0 \\ 0 & 2. & 0 & 0 & 0 & 0 \\ 0 & 0 & 3. & 0 & 0 & 0 \\ 0 & 0 & 0 & 4. & 0 & 0 \\ 0 & 0 & 0 & 0 & 5. & 0 \\ 0 & 0 & 0 & 0 & 0 & 6. \end{bmatrix}$$

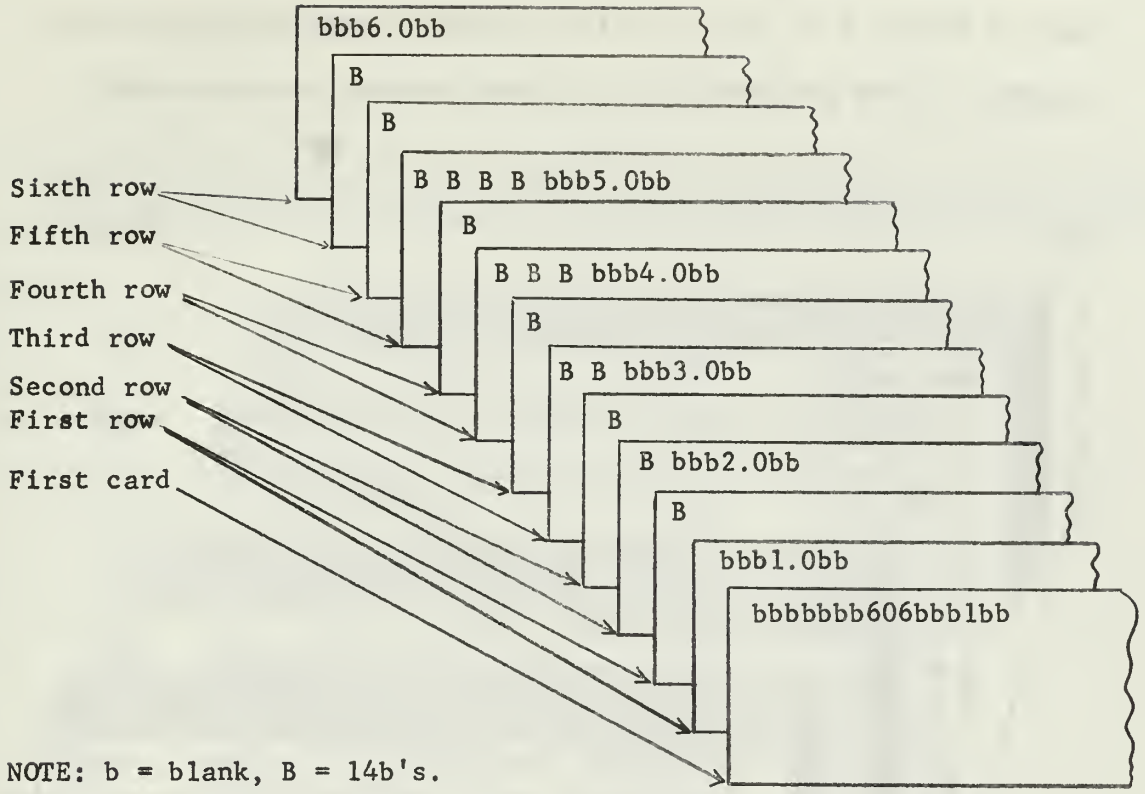

FIGURE A-1

Note that element C(1,1) is punched between fields 1 to 14, element C(2,2) between fields 15 to 28, and so on. Note too that each row has two cards because there are six elements for a row, and in each card only five consecutive elements can be punched.

If more than one set of data is used, the second set comes after the last card of the first set, starting again with the first card

bbbbbbbNNbbb1bb.....

and so on.

In order to stop the program, the final card of the data deck is punched as follows bbbbbbb0b6bbb6bb.....

The complete program deck includes: control cards, main program, subroutine package, and data deck. The assembly of these components is

shown in Figure A-2, where the first control card is from the CECI 1 program. To use program CECI 2, change the word CECI 1 to CECI 2.

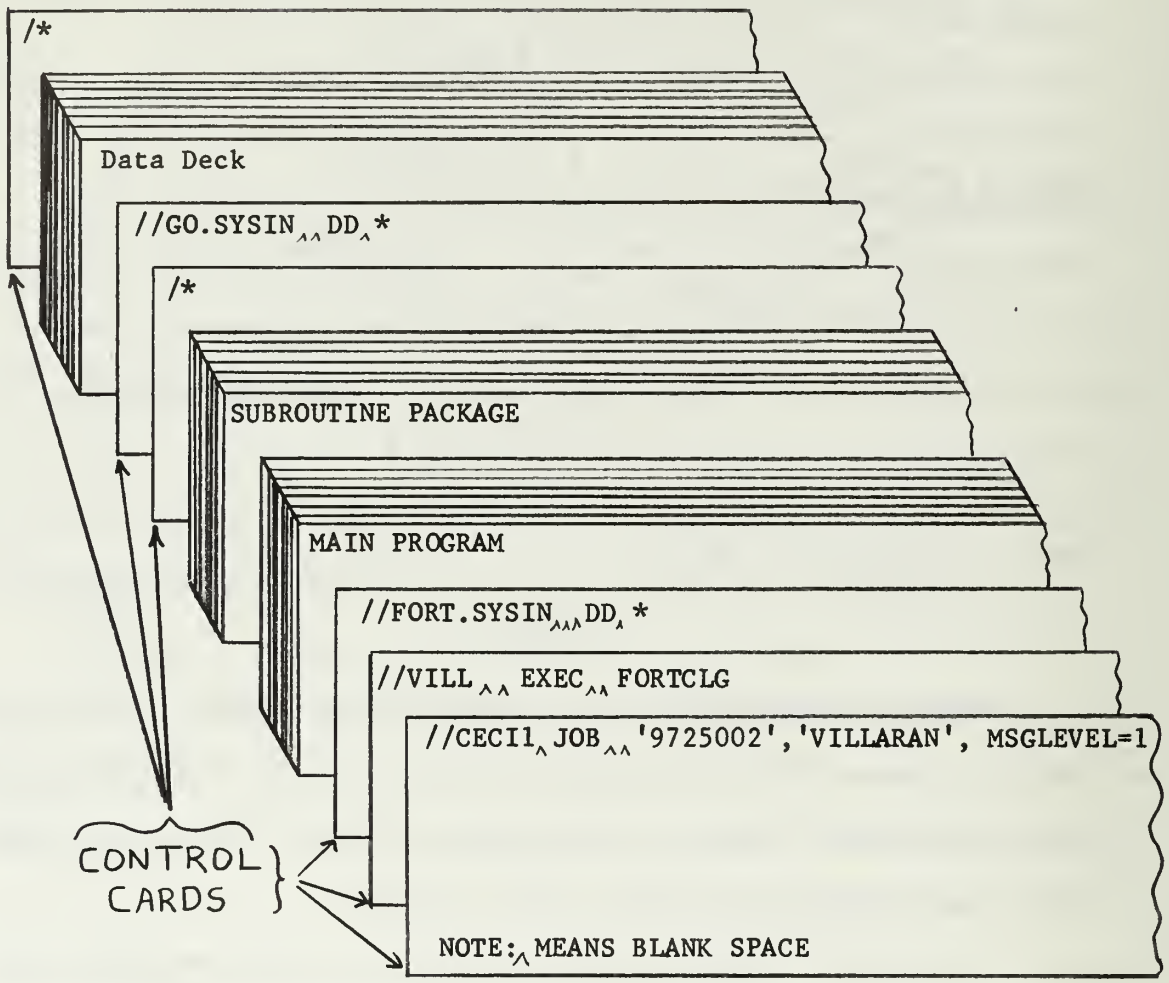

FIGURE A-2

A-2 Listing of Program CECI 1.

The cards of program CECI 1 are numbered from 0000 to 0159.

```
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                             0000
      DIMENSION T2(51,6),Z(21,21),X(21,21),AB(6,6),                     0001
     1ZZ(6,6),BE(7,50),EB(7,50),BOOT(7,6),U(7,50),V(7,50),TR(51,6),     0002
     2P(7,6),Y(7,6),PI(7,6),ROOT(7,6),W(22,21),S(22,21),COL(22),AA(22), 0003
     3Q(7,6),QQ(7,6),GG(7,6)                                            0004
      COMMON NUMBER                                                     0005
C                                                                       0006
C     PROGRAM CECI 1 "SOLVES" THE OVERDETERMINED LINEAR SYSTEM XU=V,    0007
C     WHERE X IS AN UNKNOWN SYMMETRIC (6,6) MATRIX AND U AND V ARE GIVEN 0008
C     RECTANGULAR MATRICES. THREE SOLUTIONS ARE GIVEN, WHICH "SOLVE" THE 0009
C     PROBLEM IN THE SENSE OF SATISFYING EACH OF THREE ERROR CRITERIA.  0010
C     FOR DETAILS SEE NAVAL POSTGRADUATE SCHOOL THESIS,"DETERMINATION   0011
C     OF OPTIMAL COMPLIANCE AND STIFFNESS MATRICES FROM EXPERIMENTAL    0012
C     DATA", BY LT. C. F. VILLARAN, 1967.  CARD NUMBER 0043  IS EITHER  0013
C     MODE=1 OR MODE=2.  IF MODE=2, THE PROGRAM READS INPUT MATRICES U  0014
C     AND V FROM PUNCHED CARDS ACCORDING TO FORMAT (5F14.6). READING IS 0015
C     BY COMPLETE ROW, A NEW ROW STARTING ON A NEW CARD. CARDS FOR MA-  0016
C     TRIX V FOLLOW THOSE FOR MATRIX U. THE FIRST CARD FOR EACH MATRIX  0017
C     IS AS FOLLOWS:  BBBBBBB6NNBBBM  WHERE B DENOTES BLANK, NN IS THE  0018
C     TWO DIGIT NUMBER OF COLUMNS IN U AND V, NN NOT GREATER THAN 50,   0019
C     AND M = 1 FOR MATRIX U, M = 2 FOR MATRIX V. CARDS FOR MATRICES U  0020
C     AND V INMEDIATELY FOLLOW THE  FINAL TWO CONTROL CARDS (/* AND     0021
C     //GO.SYSIN DD *) IN THE PROGRAM. IF THERE IS MORE THAN ONE SET OF 0022
C     INPUTS, THESE ARE PLACED IN ORDER FOLLOWING THE FIRST SET. THE    0023
C     FINAL CARD OF THE DATA DECK IS BBBBBBB0B6BBB6. IF MODE = 2, AS A- 0024
C     BOVE, THE PROGRAM IS DEALING WITH AN ACTUAL INPUT AND PRODUCING   0025
C     OPTIMAL OUTPUT. HOWEVER, FOR TEST AND RESEARCH PURPOSES, IT MAY BE 0026
C     USEFUL TO USE MODE = 1, IN WHICH DATA ARE GENERATED AND RANDOMLY  0027
C     MODIFIED SO AS TO PROVIDE PROBLEMS FOR SOLUTION. DETAILS OF THIS  0028
C     USE MAY BE FOUND IN THE THESIS REFERRED TO ABOVE.                 0029
C                                                                       0030
C                                                                       0031
C     MOST MATRICES ARE REGARDED AS DOUBLE SUBSCRIPT ARRAYS AS FOLLOWS; 0032
C     IF THERE ARE M ROWS AND N COLUMNS, THE ARRAY HAS DIMENSION (M+1,N) 0033
C     LET E(I,J) REFER TO THE ARRAY AND LET A(I,J) REFER TO THE MATRIX, 0034
C     THEN E(1,1) IS AN INTEGER IDENTIFYING THE ARRAY AND THE MATRIX,   0035
C     E(1,2) AND E(1,3) ARE  INTEGERS M AND N, RESPECTIVELY. FOR THE    0036
C     REMAINDER OF THE ELEMENTS, E(I+1,J) = A(I,J), THE OTHER  MATRI-   0037
C     CES THAT HAVE DIMENSIONS (N,N) ARE USED HERE AS SCRATCH MATRICES. 0038
C                                                                       0039
      CLOCK=ITIME(0)*.01                                                0040
    5 NUMBER=0                                                          0041
      CLUCK=ITIME(1)*.01                                                0042
      MODE=1                                                            0043
      IF(MODE-1)1000,1000,1250                                          0044
 1000 CALL REDEM(P,1)                                                   0045
      WRITE(6,50)                                                       0046
   50 FORMAT(19H1  MATRIX P FOLLOWS)                                    0047
```

```
      CALL PRINM(P,6)                                                    0048
      CALL REDEM(BE,2)                                                   0049
      WRITE(6,55)                                                        0050
   55 FORMAT(20H0  MATRIX BE FOLLOWS)                                    0051
      MM=BE(1,3)+.1                                                      0052
      LL=MM+1                                                            0053
      CALL PRINM(BE,MM)                                                  0054
      CALL ULTMA(EB,P,BE,7,MM,6,7)                                       0055
      WRITE(6,60)                                                        0056
   60 FORMAT(20H0  MATRIX EB FOLLOWS)                                    0057
      CALL PRINM(EB,MM)                                                  0058
      CALL BLANCA(U,BE,MM)                                               0059
      CALL BLANCA(V,EB,MM)                                               0060
      GO TO 1300                                                         0061
 1250 CALL REDEM(U,1)                                                    0062
      MM=U(1,3)+.1                                                       0063
      LL=MM+1                                                            0064
      CALL REDEM(V,2)                                                    0065
 1300 CONTINUE                                                           0066
      WRITE(6,65)                                                        0067
   65 FORMAT(19H0  MATRIX U FOLLOWS)                                     0068
      CALL PRINM(U,MM)                                                   0069
      WRITE(6,70)                                                        0070
   70 FORMAT(19H0  MATRIX V FOLLOWS)                                     0071
      CALL PRINM(V,MM)                                                   0072
      CALL TRANM(TR,U,LL,MM)                                             0073
      CALL ULTMA(P,U,TR,7,6,MM,LL)                                       0074
      CALL TRANM(TR,V,LL,MM)                                             0075
      CALL ULTMA(Q,V,TR,7,6,MM,LL)                                       0076
      CALL ULTMA(RCOT,U,TR,7,6,MM,LL)                                    0077
      CALL TRANM(GG,ROOT,7,6)                                            0078
      CALL ADDEM(Y,GG,ROOT,1.D0,7,6)                                     0079
      CALL BIGSM(W,P)                                                    0080
      NT=W(1,2)+.1                                                       0081
      NL=NT+1                                                            0082
      CALL AINVM(S,W,Z,X,NL,NT)                                          0083
      CALL BIGCM(COL,Y)                                                  0084
      CALL ULTMB(AA,S,COL,NL,NT)                                         0085
      CALL SQUAM(QQ,AA)                                                  0086
      CLOCK=ITIME(0)*.01-CLOCK                                           0087
      WRITE(6,2000)CLOCK                                                 0088
 2000 FORMAT(18H0  ELAPSED TIME = ,F12.2)                                0089
      CLOCK=ITIME(0)*.01                                                 0090
      WRITE(6,105)                                                       0091
  105 FORMAT(42H0  OUTPUT NO. 1,MINIMIZES V-SQUARE ERROR  )              0092
      CALL AINVM(GG,QQ,AB,ZZ,7,6)                                        0093
      CALL OUTPU(QQ,GG,U,V,BE,EB,TR,T2,LL,MM)                            0094
      CALL BIGSM(W,Q)                                                    0095
```

```
      CALL AINVM(S,W,X,Z,NL,NT)                                          0096
      CALL BIGCM(CCL,Y)                                                  0097
      CALL ULTMB(AA,S,COL,NL,NT)                                         0098
      CALL SQUAM(Y,AA)                                                   0099
      CALL AINVM(PI,Y,AB,ZZ,7,6)                                         0100
      CLOCK=ITIME(0)*.01-CLOCK                                           0101
      WRITE(6,2100)CLOCK                                                 0102
 2100 FORMAT(18H0  ELAPSED TIME = ,F12.2)                                0103
      CLOCK=ITIME(0)*.01                                                 0104
      WRITE(6,106)                                                       0105
  106 FORMAT(42H0  OUTPUT NO. 2, MINIMIZES U-SQUARE ERROR    )           0106
      CALL OUTPU(PI,Y,U,V,BE,EB,TR,T2,LL,MM)                             0107
      CALL ULTMA(Y,Q,P,7,6,6,7)                                          0108
      CALL ULTMA(Y,Q,P,7,6,6,7)                                          0109
      CALL ULTMA(BCOT,PI,P,7,6,6,7)                                      0110
      CALL SQROOT(ROOT,BOOT,Y)                                           0111
      CALL AINVM(QQ,P,AB,ZZ,7,6)                                         0112
      CALL ULTMA(Y,ROOT,QQ,7,6,6,7)                                      0113
      CALL AINVM(RCOT,Y,AB,ZZ,7,6)                                       0114
      CLOCK=ITIME(0)*.01-CLOCK                                           0115
      WRITE(6,2200)CLOCK                                                 0116
 2200 FORMAT(18H0  ELAPSED TIME = ,F12.2)                                0117
      CLOCK=ITIME(0)*.01                                                 0118
      WRITE(6,104)                                                       0119
  104 FORMAT(42H0  OUTPU NO.3A, MINIMIZES TRIANGLE ERROR   )             0120
      CALL OUTPU(Y,ROOT,U,V,EB,BE,T2,TR,LL,MM)                           0121
      CALL ULTMA(GG,P,Q,7,6,6,7)                                         0122
      CALL ULTMA(BCOT,P,PI,7,6,6,7)                                      0123
      CALL SQROOT(ROOT,BOOT,GG)                                          0124
      CALL ULTMA(GG,QQ,ROOT,7,6,6,7)                                     0125
      CALL AINVM(ROOT,GG,AB,ZZ,7,6)                                      0126
      CLOCK=ITIME(0)*.01-CLOCK                                           0127
      WRITE(6,2300)CLOCK                                                 0128
 2300 FORMAT(18H0  ELAPSED TIME = ,F12.2)                                0129
      CLOCK=ITIME(0)*.01                                                 0130
      WRITE(6,300)                                                       0131
  300 FORMAT(42H0  OUTPUT NO.3B, MINIMIZES TRIANGLE ERROR   )            0132
      CALL OUTPU(GG,ROOT,U,V,EB,BE,T2,TR,LL,MM)                          0133
      DO 400 I=2,7                                                       0134
      DO 400 J=1,6                                                       0135
      BOOT(I,J)=0.5D0*(Y(I,J)+GG(I,J))                                   0136
  400 CONTINUE                                                           0137
      CALL AINVM(RCOT,BOOT,AB,ZZ,7,6)                                    0138
      CLOCK=ITIME(0)*.01-CLOCK                                           0139
      WRITE(6,2400)CLOCK                                                 0140
 2400 FORMAT(18H0  ELAPSED TIME = ,F12.2)                                0141
      CLOCK=ITIME(0)*.01                                                 0142
      WRITE(6,500)                                                       0143
```

```
  500 FORMAT(42HO  OUTPUT NO.3C, MINIMIZES TRIANGLE ERROR  )                0144
      CALL OUTPU(BCOT,ROOT,U,V,EB,BE,T2,TR,LL,MM)                           0145
      CALL SQRERT(Y,P,Q)                                                    0146
      CALL AINVM(PI,Y,AB,ZZ,7,6)                                            0147
      WRITE(6,200)                                                          0148
  200 FORMAT(42HO  OUTPUT NO.3D, MINIMIZES TRIANGLE ERROR  )                0149
      CALL OUTPU(Y,PI,U,V,EB,BE,T2,TR,LL,MM)                                0150
      CLOCK=ITIME(C)*.01-CLOCK                                              0151
      WRITE(6,2500)CLOCK                                                    0152
 2500 FORMAT(18HO  ELAPSED TIME = ,F12.2)                                   0153
      CLOCK=ITIME(C)*.01                                                    0154
      CLUCK=ITIME(1)*.01-CLUCK                                              0155
      WRITE(6,3000)CLUCK                                                    0156
 3000 FORMAT(24HO  TOTAL ELAPSED TIME = ,F12.2)                             0157
      GO TO 5                                                               0158
      END                                                                   0159
```

A-3 Listing of Program CECI 2.

The cards of program CECI 2 are numbered from 00000 to 01570.

```
00000
00012
00022
00036
00040
00056
00066
00078
00086
00100
00112
00122
00130
00140
00156
00170
00180
00190
00210
00220
00230
00240
00250
00260
00277
00282
00290
00300
00310
00320
00330
00340
00350
00360
00370
00380
00390
00400
00410
00420
00430
00440
00450
00460
00470
```

```
PROGRAM CECI 2 "SOLVES" THE OVERDETERMINED LINEAR SYSTEM XU=V,
WHERE X IS AN UNKNOWN SYMMETRIC (6,6) MATRIX AND U AND V ARE GIVEN
RECTANGULAR MATRICES. ONE SOLUTION IS GIVEN, WHICH SOLVES THE
PROBLEM IN THE SENSE OF SATISFYING AN ERROR CRITERION FOR DE-
TAILS SEE NAVAL POSTGRADUATE SCHOOL THESIS, " DETERMINATION OF
OPTIMAL COMPLIANCE AND STIFFNESS MATRICES FROM EXPERIMENTAL DATA "
BY LT. C. F. VILLARAN, 1967. THE PROGRAM READS INPUT MATRICES U
AND V FROM PUNCHED CARDS ACCORDING TO FORMAT (5F14.6). READING IS
BY COMPLETE ROW, A NEW ROW STARTING ON A NEW CARD. CARDS FOR MA-
TRIX V FOLLOW THOSE FOR MATRIX U. THE FIRST CARD FOR EACH MATRIX
IS AS FOLLOWS: BRRBBB36NN3BBM WHERE B DENOTES BLANK, NN IS THE
TWO DIGIT NUMBER OF COLUMNS IN U AND V, NN NOT GREATER THAN 50,
AND M = 1 FOR MATRIX U, M = 2 FOR MATRIX V. CONTROL CARDS (/* AND
//GO.SYSIN DD *) IN THE PROGRAM. IF THERE IS MORE THAN ONE SET OF
INPUTS, THESE ARE PLACED IN ORDER FOLLOWING THE FIRST SET. AS A-
BOVE, THE PROGRAM IS DEALING WITH AN ACTUAL INPUT AND PRODUCING
OPTIMAL OUTPUT. HOWEVER, FOR TEST DATA ARE GENERATED AND, RANDOMLY
MODIFIED SO AS TO PROVIDE PROBLEMS FOR SOLUTION. DETAILS OF THIS
USE MAY BE FOUND IN THE THESIS REFERRED TO ABOVE.

MOST MATRICES ARE REGARDED AS DOUBLE SUBSCRIPT ARRAYS AS FOLLOWS;
IF THERE ARE M ROWS AND N COLUMNS, THE ARRAY HAS DIMENSION (M+1,N)
LET E(I,J) REFER TO THE ARRAY AND LET A(I,J) REFER TO THE MATRIX,
THEN E(1,1) IS AN INTEGER IDENTIFYING THE ARRAY AND THE MATRIX,
E(1,2) AND E(1,3) ARE INTEGERS M AND N, RESPECTIVELY. FOR THE
REMAINDER OF THE ELEMENTS, E(I+1,J) = A(I,J), THE OTHER MATRI-
CES THAT HAVE DIMENSIONS (N,N) ARE USED HERE AS SCRATCH MATRICES.

CARD NUMBER 0087C IS AN SCALAR QUANTITY CALLED BETA THAT DESIGNA-
TE THE RELATIVE IMPORTANCE OF THE TWO SEPARATE CRITERIA OF WHICH
ALFA 4 IS COMPOSED.

      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)
      DIMENSION T2(51,6),Z(21,21),X(21,21),AR(6,6),ZZ(6,6),BE(7,50),
     1EB(7,50),U(7,50),V(7,50),TR(51,6),P(7,6),Y(7,5),ROOT(7,6),Q(7,6),
     2W(22,21),S(22,21),COL(22),AA(22),QQ(7,6),GG(7,6),PP(7,6),R(7,6),
     3RR(7,6),T(7,6),TT(7,6),PI(7,6),SS(22,21)
      COMMON NUMBER
      CLOCK=ITIME(0)*.01
    5 NUMBER=0
      CLOCK=ITIME(1)*.01
      MODE=1
```

```
      IF(MODE-1)1000,1000,1250                                          00480
 1000 CALL REDEM(P,1)                                                   00490
      WRITE(6,50)                                                       00500
   50 FORMAT(19H1  MATRIX P FOLLOWS)                                    00510
      CALL PRINM(P,6)                                                   00520
      CALL REDEM(BE,2)                                                  00530
      WRITE(6,55)                                                       00540
   55 FORMAT(20HC  MATRIX BE FOLLOWS)                                   00550
      MM=BE(1,3)+.1                                                     00560
      LL=MM+1                                                           00570
      CALL PRINM(BE,MM)                                                 00580
      CALL ULTMA(EB,P,BE,7,MM,6,7)                                      00590
      WRITE(6,60)                                                       00600
   60 FORMAT(20HC  MATRIX EB FOLLOWS)                                   00610
      CALL PRINM(EB,MM)                                                 00620
      CALL BLANCA(U,BE,MM)                                              00630
      CALL BLANCA(V,EB,MM)                                              00640
      GO TO 1300                                                        00650
 1250 CALL REDEM(U,1)                                                   00660
      MM=U(1,3)+.1                                                      00670
      LL=MM+1                                                           00680
      CALL REDEM(V,2)                                                   00690
 1300 CONTINUE                                                          00700
      WRITE(6,65)                                                       00710
   65 FORMAT(19HC  MATRIX U FOLLOWS)                                    00720
      CALL PRINM(U,MM)                                                  00730
      WRITE(6,70)                                                       00740
   70 FORMAT(19HC  MATRIX V FOLLOWS)                                    00750
      CALL PRINM(V,MM)                                                  00760
      CALL TRANM(TR,U,LL,MM)                                            00770
      CALL ULTMA(P,U,TR,7,6,MM,LL)                                      00780
      CALL TRANM(TR,V,LL,MM)                                            00790
      CALL ULTMA(G,V,TR,7,6,MM,LL)                                      00800
      CALL ULTMA(ROOT,U,TR,7,6,MM,LL)                                   00810
      CALL TRANM(GG,ROOT,7,6)                                           00820
      CALL ADDEM(Y,GG,ROOT,1.D0,7,6)                                    00830
      CALL BIGSM(W,P)                                                   00840
      NT=W(1,2)+.1                                                      00850
      NL=NT+1                                                           00860
      BETA=0.800                                                        00870
      CALL AINVM(S,W,Z,X,NL,NT)                                         00880
      CALL BIGCM(CCL,Y)                                                 00890
      CALL ULTMB(AA,S,COL,NL,NT)                                        00900
      CALL SQUAM(GG,AA)                                                 00910
      CALL BIGSM(W,Q)                                                   00920
      CALL AINVM(SS,W,X,Z,NL,NT)                                        00930
      CALL ULTMB(AA,SS,COL,NL,NT)                                       00940
      CALL SQUAM(GG,AA)                                                 00950
```

```
      CALL AINVM(PI,GG,AB,ZZ,7,6)                                       00960
      TETA=1.D-1                                                        00970
      DO 80 I=2,7                                                       00980
      DO 80 J=1,6                                                       00990
      QQ(I,J)=(QQ(I,J)+TETA*PI(I,J))/(1.D0+TETA)                        01000
   80 CONTINUE                                                          01010
      CLOCK=ITIME(C)*.C1-CLOCK                                          01020
      WRITE(6,2000)CLOCK                                                01030
 2000 FORMAT(18HO  ELAPSED TIME = ,F12.2)                               01040
      CLOCK=ITIME(C)*.C1                                                01050
      RCL=0.D0                                                          01060
  201 DO 235 K=1,1C0                                                    01070
      TOL=K                                                             01080
      CALL AINVM(GG,QQ,AB,ZZ,7,6)                                       01090
      CALL ULTMA(PP,Q,GG,7,6,6,7)                                       01100
C                                                                       01110
      CALL ULTMA(R,GG,Q,7,6,6,7)                                        01120
      CALL ADDEM(RR,PP,R,1.D0,7,6)                                      01130
      CALL ADDEM(PP,RR,Y,-1.D0,7,6)                                     01140
      CALL ULTMA(R,GG,PP,7,6,6,7)                                       01150
      CALL ULTMA(RR,R,GG,7,6,6,7)                                       01160
      CALL ADDEM(TT,Y,RR,TETA,7,6)                                      01170
      CALL BIGCM(CCL,TT)                                                01180
      CALL ULTMB(AA,S,COL,NL,NT)                                        01190
      CALL SQUAM(PP,AA)                                                 01200
      CALL ADDEM(R,QQ,PP,-1.D0,7,6)                                     01210
C                                                                       01220
      DO 210 I=2,7                                                      01230
      DO 210 J=1,6                                                      01240
      RET=DMAX1(DABS(R(I,J)))                                           01250
  210 CONTINUE                                                          01260
      DO 220 I=2,7                                                      01270
      DO 220 J=1,6                                                      01280
      QQ(I,J)=PP(I,J)                                                   01290
  220 CONTINUE                                                          01300
      IF(RET-1.D-5)240,240,235                                          01310
  235 CONTINUE                                                          01320
C                                                                       01330
  240 RCL=RCL+TCL                                                       01340
      TETA=TETA+1.D-1                                                   01350
      IF(BETA-TETA)202,203,203                                          01360
  203 GO TO 201                                                         01370
  202 CONTINUE                                                          01380
      CALL AINVM(GG,QQ,AB,ZZ,7,6)                                       01390
      CLOCK=ITIME(C)*.01-CLOCK                                          01400
      WRITE(6,2100)CLOCK                                                01410
 2100 FORMAT(18HO  ELAPSED TIME = ,F12.2)                               01420
      CLOCK=ITIME(C)*.C1                                                01430
```

```
      WRITE(6,238)ROL                                                    01440
  238 FORMAT(26HC NUMBER OF ITERACTIONS = , D14.8)                       01450
      WRITE(6,105)                                                       01460
  105 FORMAT(53HO  OUTPUT NO.4, MINIMIZES V + BETA * U SQUARE ERRORS  )  01470
      CALL OUTPUT(QQ,GG,U,V,BE,EB,TR,T2,LL,MM,BETA)                      01480
      CLOCK=ITIME(0)*.01-CLOCK                                           01490
      WRITE(6,2200)CLOCK                                                 01500
 2200 FORMAT(18HO  ELAPSED TIME = ,F12.2)                                01510
      CLOCK=ITIME(0)*.01                                                 01520
      CLUCK=ITIME(1)*.01-CLUCK                                           01530
      WRITE(6,3000)CLUCK                                                 01540
 3000 FORMAT(24HO  TOTAL ELAPSED TIME = ,F12.2)                          01550
      GO TO 5                                                            01560
      END                                                                01570
```

APPENDIX B

SUBROUTINE PROGRAMS

B-1 Description of Subroutines.

SUBROUTINE ADDEM (A,B,C,D,M,L) provides the result of A = B±DC, where D is a scalar constant, and A, B, and C are double precision matrices dimensioned (M,L), where M and L are not less than 3. With D = 1.0, A = B+C; with D = -1.0, A = B - C.

The matrices B and C from the calling program must have identical order, otherwise a message is printed and the execution is halted.

SUBROUTINE AINVM (A,B,C,D,L,N) forms the inverse $A = B^{-1}$ of input matrix B. Matrices A, B, C, and D are double precision, with A and B dimensioned (L,N). Matrices C and D, which are used as scratch matrices, have dimension (N,N), where L = N+1, and N is not less than 3.

A message is printed if the input matrix B is non-square and the execution is halted.

SUBROUTINE BIGCM (COL,Q) forms the 21-element vector COL from the 6x6 matric Q, according to the scheme described in Section 3; if the input matrix Q is non-square a message is printed and the execution is halted.

SUBROUTINE BIGSM (BIG,P) forms the 21 x 21 matrix BIG, from the 6x6 matrix P according to the scheme described in Section 3; if the input matrix P is non-square a message is printed and the execution is halted.

SUBROUTINE BLANCA (CO,BE,N) forms matrix CO from matrix BE by introducing random errors in the latter according to the method described in Section 7. The arrays CO and BE are dimensioned (7,N). The maximum relative error, denoted by the symbol r in the exposition in Section 7, is governed by the computer variable PERC, which is one tenth the relative error. If PERC = 1.D-2 = .0100..., then r = .100..., and a maximum of

10% error is introduced. PERC may be changed by changing card number 0313 in the subroutine.

SUBROUTINE JACVAT (B,N,NOYES, RR,SS,EIRU,NDIM) gives the eigenvalues and eigenvectors of the symmetric matrix B. If B is non-symmetrical the subroutine first makes B symmetric by forming $\frac{1}{2}B+\frac{1}{2}B^T$. RR is the spectral matrix that contains the eigenvalues of B. SS is the modal matrix of B that contains the eigenvectors. EIRU is the square root of the elements in the diagonal matrix RR. N is the dimension of B. NOYES is 0 or 1; if 0, no eigenvalues will be recovered; if 1, the eigenvalues will be recovered in matrix RR. In the application made in this thesis, NDIM is the same as N. All the matrices and variables are in double precision.

SUBROUTINE MAUSS3 (N,EP,A,X,KER) is a double precision FORTRAN IV language subroutine which inverts a symmetric matrix. A is the matrix to be inverted, X is its inverse. Both have dimension of (N,N). EP is a tolerable error, taken as 1.D-20 in the present work. KER is an output flag having the value 1, if all has gone well, or 2, if there has been any difficulty of execution.

SUBROUTINE OUTPU (R,B,U,V,T1,T2,T3,T5,LL,L) has inputs R, B, U, V, where R is the desired compliance matrix . B is the desired stiffness matrix, and U, V are the data for the problem (such that RU = V, approximately). It prints the matrix R, and computes and prints the three criterion numbers α_i (i = 1,2,3). Matrices T1-T5 that appear in the calling statement are scratch matrices. LL and L are dummy variables used in the dimension of the matrices and are related to the number n of vector pairs used.

SUBROUTINE PRINM (A,L) causes the (6,L) matrix A to be printed, where L is not less than 3. The first line printed contains the following information: MATRIX NUMBER XX, YY ROWS, ZZ COLUMNS. This is

followed by a print of the matrix, line by line, in FORMAT (1X, 8D12.4).

SUBROUTINE RECIP (A,B) A and B are diagonal matrices (6,6) and $A = B^{-1}$, where B is the input matrix.

SUBROUTINE REDEM (A,IDENT) reads into the computer memory from the data deck which is punched as described in Appendix A. A is the "name" of the matrix and IDENT is an identifying integer to assure that the card-group which is read is indeed the one desired.

SUBROUTINE SQRERT (A,B,C) solves the matrix equation that satisfies criterion 3, using the procedure that twice takes the square root of a symmetric matrix. $B = UU^T$, $C = VV^T$ are the inputs, and A is the desired compliance matrix.

SUBROUTINE SQROOT (A,CC,G) takes the square root of a non-symmetric matrix by iteration procedure. Matrices CC and G are inputs. CC is the approximate square root; G is the matrix whose square root is desired, and A is the desired square root.

SUBROUTINE SQUAM (A,B) forms a 6 by 6 matrix A from the 21-element column vector B, arranging the results properly as implied in Section 3.

SUBROUTINE TRACM (T,A) forms the trace T of the 6 by 6 matrix A.

SUBROUTINE TRANM (A,B,LL,L) forms the transpose of matrix B(7,L), obtaining matrix A(LL,6), where L is no less than 3, and LL = L + 1.

SUBROUTINE ULTMA (A,B,C,MM,M,L,LL) forms the matrix product A = BC. A(MM,M) = B(MM,L) x C(LL,M), where MM, M,L,LL, can be any numbers not less than 3 for which the matrices are conformable for multiplication. A message is printed if the matrices B and C are not conformable and execution is halted.

SUBROUTINE ULTMB (A,B,C,LL,L) multiplies the square matrix B by a column matrix C, to obtain the column matrix A. A(LL) = B(LL,L) x C(LL),

where L is any number not less than 3, and LL = L + 1.

SUBROUTINE OUTPUT (R,B,U,V,T1,T2,T3,T5,LL,L,BETA) is the same as SUBROUTINE OUTPU. The only differences are the inclusion of parameter BETA in the calling statement, and the printing of four criterion numbers instead of three, α_i (i = 1,2,3,4).

B-2 Subroutine Listing

	CARDS NUMBERED	
SUBROUTINE	FROM	TO
ADDEM	0160	0182
AINVM	0183	0208
BIGCM	0209	0233
BIGSM	0234	0288
BLANCA	0289	0336
JACVAT	0337	0481
MAUSS3	0482	0532
OUTPU	0533	0559
PRINM	0560	0576
RECIP	0577	0594
REDEM	0595	0616
SQRERT	0617	0638
SQROOT	0639	0692
SQUAM	0693	0727
TRACM	0728	0744
TRANM	0745	0763
ULTMA	0764	0791
ULTMB	0792	0814
OUTPUT	20000	20360

```
C  SUBROUTINE ADDEM ADD MXL MATRICES, WHERE M=7,50 AND L=6,50                0160
      SUBROUTINE ADDEM (A,B,C,D,M,L)                                         0161
      REAL*8 A,B,C,D                                                         0162
      DIMENSION A(M,L),B(M,L),C(M,L)                                         0163
      COMMON NUMBER                                                          0164
      NUMBER=NUMBER+1                                                        0165
      A(1,1)=NUMBER                                                          0166
      IF(DABS(B(1,2)-C(1,2))-.1D0)1003,1002,1002                             0167
 1003 IF(DABS(B(1,3)-C(1,3))-.1D0)1004,1002,1002                             0168
 1004 A(1,2)=B(1,2)                                                          0169
      A(1,3)=B(1,3)                                                          0170
      M1=A(1,2)+1.1D0                                                        0171
      N=A(1,3)+.1D0                                                          0172
      DO 1005 J=2,M1                                                         0173
      DO 1005 K=1,N                                                          0174
      A(J,K)=B(J,K)+D*C(J,K)                                                 0175
 1005  CONTINUE                                                             0176
      GO TO 3334                                                             0177
 1002 WRITE(6,1006) NUMBER                                                   0178
 1006 FORMAT(48H  NONCONFORMABLE MATRIX ADDITION TO FORM NUMBER, I7)         0179
      STOP                                                                   0180
 3334 RETURN                                                                 0181
      END                                                                    0182

C     SUBROUTINE AINVM INVERTS N X N MATRIX                                  0183
      SUBROUTINE AINVM(A,B,C,D,L,N)                                          0184
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                                  0185
      DIMENSION A(L,N),B(L,N),C(N,N),D(N,N)                                  0186
      COMMON NUMBER                                                          0187
      NUMBER=NUMBER+1                                                        0188
      IF(DABS(B(1,2)-B(1,3))-1.D-1)1025,1023,1023                            0189
 1023 WRITE(6,1024) NUMBER                                                   0190
 1024 FORMAT(37H  NONSQUARE INVERSION TO FORM NUMBER , I7)                   0191
 3333 STOP                                                                   0192
 1025 A(1,1)=NUMBER                                                          0193
      A(1,2)=B(1,2)                                                          0194
      A(1,3)=B(1,2)                                                          0195
      DO 1026 J=1,N                                                          0196
      DO 1026 K=1,N                                                          0197
      M=J+1                                                                  0198
      D(J,K)=B(M,K)                                                          0199
 1026 CONTINUE                                                               0200
      CALL MAUSS3(N,1.D-20,D,C,K2)                                           0201
      DO 1027 J=2,L                                                          0202
      DO 1027 K=1,N                                                          0203
      JJ=J-1                                                                 0204
      A(J,K)=C(JJ,K)                                                         0205
 1027 CONTINUE                                                               0206
      RETURN                                                                 0207
      END                                                                    0208
```

```
C       SUBROUTINE BIGCM FORMS BIG COLUMN MATRIX,COL,FROM SQUARE MATRIX,Q.
        SUBROUTINE BIGCM(COL,Q)
        REAL*8 COL,Q,DELTA
        DIMENSION COL(22),Q(7,6)
        COMMON NUMBER
        NUMBER=NUMBER+1
        IF(DABS(Q(1,2)-Q(1,3))-1.D-1)722,720,720
  720   WRITE(6,721)NUMBER
  721   FORMAT(39H  NONSQUARE INPUT TO FORM BIGCM NUMBER , I5)
        STOP
  722   N=Q(1,2)+.1
        JJJ=1
        DO 723 K=1,N
        DO 723 J=1,K
        JJJ=JJJ+1
        JJ=J+1
        IF(J-K)724,725,724
  724   DELTA=1.D0
        GO TO 726
  725   DELTA=2.D0
  726   COL(JJJ)=Q(JJ,K)/DELTA
  723   CONTINUE
        COL(1)=JJJ-1
        RETURN
        END
```

```
C       SUBROUTINE BIGSM FORMS BIG SQUARE MATRIX, BIG, FROM SQUARE MATRIX     0234
C       P.                                                                    0235
        SUBROUTINE BIGSM(BIG,P)                                               0236
        REAL*8 BIG,P,T                                                        0237
        DIMENSION BIG(22,21),P(7,6),NA(21,21)                                 0238
        COMMON NUMBER                                                         0239
        NUMBER=NUMBER+1                                                       0240
        IF(DABS(P(1,2)-P(1,3))-1.D-1)822,820,820                              0241
  820   WRITE(6,821) NUMBER                                                   0242
  821   FORMAT(41H  NONSQUARE INPUT TO FORM BIGSUM A NUMBER, I5)              0243
        STOP                                                                  0244
  822   N=P(1,2)+.1                                                           0245
        MMM=N                                                                 0246
        NA(1,1)=1                                                             0247
        NA(1,2)=2                                                             0248
        DO 50 J=3,MMM                                                         0249
        JM=J-1                                                                0250
        JMM=J-2                                                               0251
        NA(1,J)=1+2*NA(1,JM)-NA(1,JMM)                                        0252
   50   CONTINUE                                                              0253
        DO 70 J=2,MMM                                                         0254
        DO 60 K=2,J                                                           0255
        KM=K-1                                                                0256
        NA(K,J)=NA(KM,J)+1                                                    0257
        NA(J,K)=NA(K,J)                                                       0258
   60   CONTINUE                                                              0259
   70   CONTINUE                                                              0260
        DO 71 J=2,MMM                                                         0261
   71   NA(J,1)=NA(1,J)                                                       0262
        NT=N*(N+1)                                                            0263
        T=NT                                                                  0264
        NP1=N+1                                                               0265
        NNNN=0.5D0*T+1.D-3                                                    0266
        NNN1=NNNN+1                                                           0267
        DO 824 I=1,NNN1                                                       0268
        DO 1824 J=1,NNNN                                                      0269
        BIG(I,J)=0.DC                                                         0270
 1824   CONTINUE                                                              0271
  824   CONTINUE                                                              0272
        BIG(1,1)=NUMBER                                                       0273
        BIG(1,2)=.5DC*T                                                       0274
        BIG(1,3)=BIG(1,2)                                                     0275
        NP=N+1                                                                0276
        DO 823 I=2,NP                                                         0277
        IM=I-1                                                                0278
        DO 1823 J=1,N                                                         0279
        DO 2823 K=1,N                                                         0280
        IP=NA(K,IM)+1                                                         0281
```

```
      JP=NA(K,J)                                                        0282
      BIG(IP,JP)=BIG(IP,JP)+P(I,J)                                      0283
 2823 CONTINUE                                                          0284
 1823 CONTINUE                                                          0285
  823 CONTINUE                                                          0286
      RETURN                                                            0287
      END                                                               0288
```

```
C  SUBROUTINE BLANCA GIVE RANDOM VALUES FOR MATRIX,  CO=OUTPUT,BE=INPUT
      SUBROUTINE BLANCA(CO,BE,N)
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)
      DIMENSION CO(7,N),BE(7,N)
      COMMON NUMBER
      NUMBER=NUMBER+1
      CO(1,1)=NUMBER
      CO(1,2)=BE(1,2)
      CO (1,3)=BE(1,3)
      NN=BE(1,3)
      RHO=NUMBER
      DO 5 I=2,7
      P=I
      DO 5 J=1,NN
      Q=J
      RHO=RHO+3.632700
      IF(RHO-9.999D0)12,11,11
   11 RHO=RHO-8.415300
   12 XX=RHO**(P/1.D01+Q/2.5D1+3.1416D0/5.D0)
      KX=XX
      AKX=KX
      BKX=XX-AKX
      CKX=1.D1*BKX
      TT=BE(I,J)
      PERC=0.D0
      QQ=PERC*CKX*TT
      KKX=CKX
      DKKX=KKX
      EKKX=DKKX/2.D0
      LKKX=EKKX
      GKKX=LKKX
      OKKX=EKKX-GKKX
      IF(TT)15,10,15
   10 RR=1.D1*PERC*CKX
      IF(OKKX-5.D-1)6,7,6
    6 CO(I,J)=RR
      GO TO 5
    7 CO(I,J)=-RR
      GO TO 5
   15 IF(OKKX-5.D-1)3,1,3
    3 CO(I,J)=TT+QQ
      GO TO 5
    1 CO(I,J)=TT-QQ
    5 CONTINUE
      WRITE(6,222) PERC
  222 FORMAT(20H0 ERROR IN BLANCA = ,F10.5 )
      RETURN
      END
```

```
C   IN JACVAT B(7,6)=A(6,6) IS THE SQUARE MATRIX,FOR OUR PORPOUSE A IS      0337
C   SYMETRIX AND POSITIVE DEFINITE, N IS THE DIMENSION OF A, NOYES=1,       0338
C   RR(7,6)=EIVU(6,6) IS THE MATRIX OF EIGENVALUES(RR=SPECTRAL),SS(7,6)=    0339
C   EIVR(6,6) IS THE MODAL MATRIX(EIGENVECTORS), EIRU(7,6) IS THE SQUARE    0340
C   ROOT OF THE ELEMENTS IN DIAGONAL EIVU, NDIM IS THE ACTUAL NUMERICAL     0341
C   VALUE OF N.                                                             0342
      SUBROUTINE JACVAT(B,N,NOYES,RR,SS,EIRU,NDIM)                          0343
      DIMENSION B(7,6),EIVU(6),RR(7,6),EIVR(6,6),EIRU(7,6),A(7,6),          0344
     1SS(7,6)                                                               0345
      REAL*8 A,EIVU,EIVR,EIRU,ATOP,AVGF,D,S,DSTOP,THRSH,AIJ,AII,AJJ,C,T,    0346
     1U,RR,B,SS                                                             0347
      COMMON NUMBER                                                         0348
      DO 200 I=1,N                                                          0349
      DO 200 J=1,N                                                          0350
      A(I,J)=B(I+1,J)                                                       0351
  200 CONTINUE                                                              0352
      IF(NOYES)99,102,99                                                    0353
   99 CONTINUE                                                              0354
      DO 101 J=1,N                                                          0355
      DO 100 I=1,N                                                          0356
  100 EIVR(I,J)=0.D0                                                        0357
  101 EIVR(J,J)=1.D0                                                        0358
  102 ATOP=0.D0                                                             0359
      DO 112 J=1,N                                                          0360
      DO 111 I=1,J                                                          0361
      IF(A(I,J)-A(J,I))90,103,90                                            0362
   90 A(I,J)=0.5D0*(A(I,J)+A(J,I))                                          0363
      A(J,I)=A(I,J)                                                         0364
  103 CONTINUE                                                              0365
      IF(ATOP-DABS(A(I,J)))104,111,111                                      0366
  104 ATOP=DABS(A(I,J))                                                     0367
  111 CONTINUE                                                              0368
  112 EIVU(J)=A(J,J)                                                        0369
      IF(ATOP)109,109,113                                                   0370
  109 WRITE(6,110)                                                          0371
  110 FORMAT(26H IN JACVAT, MATRIX A  = 0 )                                 0372
      RETURN                                                                0373
  113 AVGF=DFLOAT(N+(N-1))*5.5D-1                                           0374
      D=0.D0                                                                0375
      DO 114 JJ=2,N                                                         0376
      DO 114 II=2,JJ                                                        0377
      S=A(II-1,JJ)/ATOP                                                     0378
  114 D=S*S+D                                                               0379
      DSTOP=(1.D-14)*D                                                      0380
      THRSH=DSQRT(D/AVGF)*ATOP                                              0381
  115 IFLAG=0                                                               0382
      DO 130 JCOL=2,N                                                       0383
      JCOL1=JCOL-1                                                          0384
```

```
      DO 130 IROW=1,JCOL1                                                0385
      AIJ=A(IROW,JCOL)                                                   0386
      IF(DABS(AIJ)-THRSH)130,130,117                                     0387
  117 AII=A(IROW,IROW)                                                   0388
      AJJ=A(JCOL,JCOL)                                                   0389
      S=AJJ-AII                                                          0390
      IF (DABS(AIJ)-1.D-18*DABS(S))130,130,118                           0391
  118 IFLAG=1                                                            0392
      IF(1.D-20*DABS(AIJ)-DABS(S))116,119,119                            0393
  119 S=5.D-1                                                            0394
      S=DSQRT(S)                                                         0395
      C=S                                                                0396
      GO TO 120                                                          0397
  116 T=AIJ/S                                                            0398
      S=0.25D0/DSQRT(0.25D0+T*T)                                         0399
      C=DSQRT(0.5D0+S)                                                   0400
      S=2.D0*T*S/C                                                       0401
  120 DO 121 I=1,IROW                                                    0402
      T=A(I,IROW)                                                        0403
      U=A(I,JCOL)                                                        0404
      A(I,IROW)=C*T-S*U                                                  0405
  121 A(I,JCOL)=S*T+C*U                                                  0406
      I2=IROW+2                                                          0407
      IF(I2-JCOL)127,127,123                                             0408
  127 CONTINUE                                                           0409
      DO 122 I=I2,JCOL                                                   0410
      T=A(I-1,JCOL)                                                      0411
      U=A(IROW,I-1)                                                      0412
      A(I-1,JCOL)=S*U+C*T                                                0413
  122 A(IROW,I-1)=C*U-S*T                                                0414
  123 A(JCOL,JCOL)=S*AIJ+C*AJJ                                           0415
      A(IROW,IROW)=C*A(IROW,IROW)-S*(C*AIJ-S*AJJ)                        0416
      DO 124 J=JCOL,N                                                    0417
      T=A(IROW,J)                                                        0418
      U=A(JCOL,J)                                                        0419
      A(IROW,J)=C*T-S*U                                                  0420
  124 A(JCOL,J)=S*T+C*U                                                  0421
      IF(NOYES)131,126,131                                               0422
  131 CONTINUE                                                           0423
      DO 125 I=1,N                                                       0424
      T=EIVR(I,IROW)                                                     0425
      EIVR(I,IROW)=C*T-EIVR(I,JCOL)*S                                    0426
  125 EIVR(I,JCOL)=S*T+EIVR(I,JCOL)*C                                    0427
  126 CONTINUE                                                           0428
      S=AIJ/ATOP                                                         0429
      D=D-S*S                                                            0430
      IF(D-DSTOP)1260,129,129                                            0431
 1260 D=0.D0                                                             0432
```

```
      DO 128 JJ=2,N                                                     0433
      DO 128 II=2,JJ                                                    0434
      S=A(II-1,JJ)/ATOP                                                 0435
  128 D=S*S+D                                                           0436
      DSTOP=(1.D-14)*D                                                  0437
  129 THRSH=DSQRT(D/AVGF)*ATOP                                          0438
  130 CONTINUE                                                          0439
      IF(IFLAG)115,134,115                                              0440
  134 T=A(1,1)                                                          0441
      A(1,1)=EIVU(1)                                                    0442
      EIVU(1)=T                                                         0443
      DO 132 J=2,N                                                      0444
      T=A(J,J)                                                          0445
      A(J,J)=EIVU(J)                                                    0446
      EIVU(J)=T                                                         0447
      DO 132 I=2,J                                                      0448
  132 A(I-1,J)=A(J,I-1)                                                 0449
      DO 25 I=1,N                                                       0450
      DO 25 J=1,N                                                       0451
      SS(I+1,J)=EIVR(I,J)                                               0452
   25 CONTINUE                                                          0453
      NUMBER=NUMBER+1                                                   0454
      SS(1,1)=NUMBER                                                    0455
      SS(1,2)=N                                                         0456
      SS(1,3)=N                                                         0457
      DO 1300 I=1,7                                                     0458
      DO 1300 J=1,6                                                     0459
      RR(I,J)=0.D0                                                      0460
 1300 CONTINUE                                                          0461
      DO 1350 I=1,6                                                     0462
      RR(I+1,I)=EIVU(I)                                                 0463
 1350 CONTINUE                                                          0464
      NUMBER=NUMBER+1                                                   0465
      RR(1,1)=NUMBER                                                    0466
      RR(1,2)=N                                                         0467
      RR(1,3)=N                                                         0468
      DO 1400 I=1,7                                                     0469
      DO 1400 J=1,6                                                     0470
      EIRU(I,J)=0.D0                                                    0471
 1400 CONTINUE                                                          0472
      DO 1500 I=1,N                                                     0473
      EIRU(I+1,I)=DSQRT(RR(I+1,I))                                      0474
 1500 CONTINUE                                                          0475
      NUMBER=NUMBER+1                                                   0476
      EIRU(1,1)=NUMBER                                                  0477
      EIRU(1,2)=N                                                       0478
      EIRU(1,3)=N                                                       0479
  133 RETURN                                                            0480
      END                                                               0481
```

```
      SUBROUTINE MAUSS3(N,EP,A,X,KER)                                   0482
      REAL*8 A,X,Z,RATIO,S                                              0483
      DIMENSION A(N,N),X(N,N)                                           0484
      DO1 I=1,N                                                         0485
      DO 1 J=1,N                                                        0486
    1 X(I,J)=0.DO                                                       0487
      DO 2 K=1,N                                                        0488
    2 X(K,K)=1.DO                                                       0489
   10 DO 34 L=1,N                                                       0490
      KP=0                                                              0491
      Z=0.DO                                                            0492
      DO 12 K=L,N                                                       0493
      IF(Z-DABS(A(K,L)))11,12,12                                        0494
   11 Z=DABS(A(K,L))                                                    0495
      KP=K                                                              0496
   12 CONTINUE                                                          0497
      IF(L-KP)13,20,20                                                  0498
   13 DO 14 J=L,N                                                       0499
      Z=A(L,J)                                                          0500
      A(L,J)=A(KP,J)                                                    0501
   14 A(KP,J)=Z                                                         0502
      DO 15 J=1,N                                                       0503
      Z=X(L,J)                                                          0504
      X(L,J)=X(KP,J)                                                    0505
   15 X(KP,J)=Z                                                         0506
   20 IF(DABS(A(L,L))-EP)50,50,30                                       0507
   30 IF(L-N)31,34,34                                                   0508
   31 LP1=L+1                                                           0509
      DO 36 K=LP1,N                                                     0510
      IF(A(K,L))32,36,32                                                0511
   32 RATIO=A(K,L)/A(L,L)                                               0512
      DO 33 J=LP1,N                                                     0513
   33 A(K,J)=A(K,J)-RATIO*A(L,J)                                        0514
      DO 35 J=1,N                                                       0515
   35 X(K,J)=X(K,J)-RATIO*X(L,J)                                        0516
   36 CONTINUE                                                          0517
   34 CONTINUE                                                          0518
   40 DO 43 I=1,N                                                       0519
      II=N+1-I                                                          0520
      DO 43 J=1,N                                                       0521
      S=0.DO                                                            0522
      IF(II-N)41,43,43                                                  0523
   41 IIP1=II+1                                                         0524
      DO 42 K=IIP1,N                                                    0525
   42 S=S+A(II,K)*X(K,J)                                                0526
   43 X(II,J)=(X(II,J)-S)/A(II,II)                                      0527
      KER=1                                                             0528
      RETURN                                                            0529
   50 KER=2                                                             0530
      RETURN                                                            0531
      END                                                               0532
```

```
      SUBROUTINE OUTPU(R,B,U,V,T1,T2,T3,T5,LL,L)                        0533
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                             0534
      DIMENSION R(7,6),B(7,6),U(7,L),V(7,L),T1(7,L),T2(7,L),T3(LL,6),   0535
     1T4(7,6),O(5),T5(LL,6)                                             0536
      WRITE(6,32)                                                       0537
   32 FORMAT(19HO DESIRED MATRIX IS  )                                  0538
      CALL PRINM(R,6)                                                   0539
      CALL ULTMA(T1,R,U,7,L,6,7)                                        0540
      CALL ADDEM(T2,T1,V,-1.D0,7,L)                                     0541
      CALL TRANM(T3,T2,LL,L)                                            0542
      CALL ULTMA(T4,T2,T3,7,6,L,LL)                                     0543
      CALL TRACM(A,T4)                                                  0544
      O(1)=A                                                            0545
      CALL ULTMA(T2,B,V,7,L,6,7)                                        0546
      CALL ADDEM(T1,U,T2,-1.D0,7,L)                                     0547
      CALL TRANM(T5,T1,LL,L)                                            0548
      CALL ULTMA(T4,T1,T5,7,6,L,LL)                                     0549
      CALL TRACM(A,T4)                                                  0550
      O(2)=A                                                            0551
      CALL ULTMA(T4,T1,T3,7,6,L,LL)                                     0552
      CALL TRACM(A,T4)                                                  0553
      O(3)=A                                                            0554
      DO 30 I=1,3                                                       0555
   30 WRITE(6,31)I,O(I)                                                 0556
   31 FORMAT(20HO  CRITERION NUMBER ,I4,8H EQUALS ,1PE14.6)             0557
      RETURN                                                            0558
      END                                                               0559

C  SUBROUTINE PRINM, PRINT 6XN MATRIX, WHERE N=6,50                     0560
      SUBROUTINE PRINM(A,L)                                             0561
      REAL*8 A                                                          0562
      DIMENSION A(7,L)                                                  0563
      NT1=A(1,1)+.1                                                     0564
      NT2=A(1,2)+.1                                                     0565
      NT3=A(1,3)+.1                                                     0566
      WRITE(6,1020)NT1,NT2,NT3                                          0567
      M1=A(1,2)+1.1                                                     0568
      N=A(1,3)+.1                                                       0569
      DO 1021 J=2,M1                                                    0570
      WRITE(6,1022)(A(J,K),K=1,N)                                       0571
 1021 CONTINUE                                                          0572
 1022 FORMAT(1X,8D12.4)                                                 0573
 1020 FORMAT(18HO   MATRIX NUMBER , I7,3H,  ,I2,6H ROWS,,I2,8H COLUMNS/)  0574
      RETURN                                                            0575
      END                                                               0576
```

```
C  SUBROUTINE RECIP INVERSE A DIAGONAL MATRIX (6X6)                    0577
      SUBROUTINE RECIP(A,B)                                            0578
      REAL*8 A,B                                                       0579
      DIMENSION A(7,6),B(7,6)                                          0580
      COMMON NUMBER                                                    0581
      NUMBER=NUMBER+1                                                  0582
      A(1,1)=NUMBER                                                    0583
      A(1,2)=B(1,2)                                                    0584
      A(1,3)=B(1,3)                                                    0585
      DO 5 I=2,7                                                       0586
      DO 5 J=1,6                                                       0587
      IF(I-1-J)10,15,10                                                0588
   10 A(I,J)=0.D0                                                      0589
      GO TO 5                                                          0590
   15 A(I,J)=1.D0/B(I,J)                                               0591
    5 CONTINUE                                                         0592
      RETURN                                                           0593
      END                                                              0594

C  SUBROUTINE REDEM LOADS A MATRIX FROM CARDS                          0595
      SUBROUTINE REDEM(A,IDENT)                                        0596
      REAL*8 A                                                         0597
      DIMENSION A(7,50)                                                0598
      COMMON NUMBER                                                    0599
 1040 FORMAT(6X, 2I2,I4)                                               0600
 1041 FORMAT(5F14.6)                                                   0601
      READ(5,1040)M,N,JDENT                                            0602
      IF(IDENT-JDENT)1043,1044,1043                                    0603
 1043 WRITE(6,1045)NUMBER,IDENT,JDENT                                  0604
 1045 FORMAT(28H  READ MIXUP TO FORM NUMBER ,I7,10H  IDENT = ,I4,      0605
     110H  JDEN = , I4)                                                0606
 3333 STOP                                                             0607
 1044 M1=M+1                                                           0608
      DO 1042 J=2,M1                                                   0609
 1042 READ(5,1041) (A(J,K),K=1,N)                                      0610
      NUMBER=NUMBER+1                                                  0611
      A(1,1)=NUMBER                                                    0612
      A(1,2)=M                                                         0613
      A(1,3)=N                                                         0614
      RETURN                                                           0615
      END                                                              0616
```

```
      SUBROUTINE SQRERT(A,B,C)                                                  0617
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                                     0618
      DIMENSION A(7,6),B(7,6),C(7,6),SS(7,6),P(7,6),S(7,6),E(7,6),              0619
     1EE(7,6),D(7,6),CC(7,6),Q(7,6),G(7,6),F(7,6),TT(7,6),T(7,6),RR(7,6)        0620
      COMMON NUMBER                                                             0621
      CALL JACVAT(B,6,1,SS,P,S,6)                                               0622
      CALL ULTMA(E,P,S,7,6,6,7)                                                 0623
      CALL TRANM(EE,E,7,6)                                                      0624
      CALL ULTMA(SS,C,E,7,6,6,7)                                                0625
      CALL ULTMA(D,EE,SS,7,6,6,7)                                               0626
      CALL JACVAT(D,6,1,SS,CC,Q,6)                                              0627
      CALL ULTMA(SS,CC,Q,7,6,6,7)                                               0628
      CALL TRANM(G,CC,7,6)                                                      0629
      CALL ULTMA(F,SS,G,7,6,6,7)                                                0630
      CALL RECIP(TT,S)                                                          0631
      CALL TRANM(T,P,7,6)                                                       0632
      CALL ULTMA(S,P,TT,7,6,6,7)                                                0633
      CALL ULTMA(RR,S,F,7,6,6,7)                                                0634
      CALL ULTMA(P,RR,TT,7,6,6,7)                                               0635
      CALL ULTMA(A,P,T,7,6,6,7)                                                 0636
      RETURN                                                                    0637
      END                                                                       0638
```

```
      SUBROUTINE SQROOT(A,CC,G)                                          0639
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                              0640
      DIMENSION A(7,6),E(7,6),G(7,6),AA(7,6),CC(7,6),DD(7,6)             0641
      COMMON NUMBER                                                      0642
      FAC=G(2,1)                                                         0643
      DO 5 I=2,7                                                         0644
      DO 5 J=1,6                                                         0645
      G(I,J)=1.DO/FAC*G(I,J)                                             0646
      CC(I,J)=1.DO/DSQRT(FAC)*CC(I,J)                                    0647
    5 CONTINUE                                                           0648
      NUMBER=NUMBER+1                                                    0649
      E(1,1)=NUMBER                                                      0650
      E(1,2)=6.DO                                                        0651
      E(1,3)=6.DO                                                        0652
      NUMBER=NUMBER+1                                                    0653
      DD(1,1)=NUMBER                                                     0654
      DD(1,2)=6.DO                                                       0655
      DD(1,3)=6.DO                                                       0656
      DO 50 I=2,7                                                        0657
      DO 50 J=1,6                                                        0658
      IF(I-1-J)52,54,52                                                  0659
   54 E(I,J)=DSQRT(G(I,J))                                               0660
      AA(I,J)=0.DO                                                       0661
      DD(I,J)=CC(I,J)-E(I,J)                                             0662
      GO TO 50                                                           0663
   52 E(I,J)=0.DO                                                        0664
      AA(I,J)=G(I,J)                                                     0665
      DD(I,J)=CC(I,J)                                                    0666
   50 CONTINUE                                                           0667
      DO 75 K=1,50CC                                                     0668
  100 CALL ULTMA(G,DD,DD,7,6,6,7)                                        0669
      DO 60 I=2,7                                                        0670
      DO 60 J=1,6                                                        0671
      A(I,J)=(1.DO/(E(I,I-1)+E(J+1,J)))*(AA(I,J)-G(I,J))                 0672
   60 CONTINUE                                                           0673
      CALL ADDEM(CC,A,DD,-1.DO,7,6)                                      0674
      DO 65 I=2,7                                                        0675
      DO 65 J=1,6                                                        0676
      PT=DMAX1(CC(I,J))                                                  0677
   65 CONTINUE                                                           0678
      DO 70 I=2,7                                                        0679
      DO 70 J=1,6                                                        0680
      DD(I,J)=A(I,J)                                                     0681
   70 CONTINUE                                                           0682
      IF(DABS(PT)-1.D-10)80,80,75                                        0683
   75 CONTINUE                                                           0684
   80 CONTINUE                                                           0685
      CALL ADDEM(A,DD,E,1.DO,7,6)                                        0686
      DO 10 I=2,7                                                        0687
      DO 10 J=1,6                                                        0688
      A(I,J)=DSQRT(FAC)*A(I,J)                                           0689
   10 CONTINUE                                                           0690
      RETURN                                                             0691
      END                                                                0692
```

```
C  SUBROUTINE SQUAM FORMS SQUARE MATRIX FROM LARGE COLUMN                    0693
      SUBROUTINE SQUAM(A,B)                                                  0694
      REAL*8 A,B                                                             0695
      DIMENSION A(7,6),B(22)                                                 0696
      COMMON NUMBER                                                          0697
      NUMBER=NUMBER+1                                                        0698
      N=0                                                                    0699
      NNN=1                                                                  0700
   80 N=N+1                                                                  0701
      IF(N-100)86,85,85                                                      0702
   86 M=N*(N+1)/2                                                            0703
      MM=B(1)+.1                                                             0704
      IF(MM-M)80,81,80                                                       0705
   85 WRITE(6,84)NUMBER,NNN                                                  0706
   84 FORMAT(40H0  INTEGER ERROR IN SQUAM TO GET NUMBER ,2I7)                0707
      STOP                                                                   0708
   81 A(1,1)=NUMBER                                                          0709
      A(1,2)=N                                                               0710
      A(1,3)=N                                                               0711
      NO=1                                                                   0712
      DO 82 J=1,N                                                            0713
      DO 82 K=1,J                                                            0714
      NO=NO+1                                                                0715
      JJ=J+1                                                                 0716
      KK=K+1                                                                 0717
      A(JJ,K)=B(NO)                                                          0718
      A(KK,J)=B(NO)                                                          0719
   82 CONTINUE                                                               0720
      IF(NO-1-MM)87,88,87                                                    0721
   87 NNN=2                                                                  0722
      WRITE(6,84) NUMBER,NNN                                                 0723
      STOP                                                                   0724
   88 CONTINUE                                                               0725
      RETURN                                                                 0726
      END                                                                    0727
```

```
C SUBROUTINE TRACM COMPUTES TRACE OF SQUARE MATRIX                              0728
      SUBROUTINE TRACM(T,A)                                                    0729
      REAL*8 A,T                                                               0730
      DIMENSION A(7,6)                                                         0731
      N=A(1,1)+.1                                                              0732
      IF(DABS(A(1,2)-A(1,3))-1.D-1)680,681,681                                 0733
  681 WRITE(6,682) N                                                           0734
  682 FORMAT(49H  NONSQUARE MATRIX ARGUMENT OF TRACM, MATRIX NO. , I7)         0735
      STOP                                                                     0736
  680 M=A(1,2)+.1                                                              0737
      T=0.D0                                                                   0738
      DO 683 J=1,M                                                             0739
      JP=J+1                                                                   0740
      T=T+A(JP,J)                                                              0741
  683 CONTINUE                                                                 0742
      RETURN                                                                   0743
      END                                                                      0744


C  SUBROUTINE TRANM TRANSPOUSE MATRIX 7XN INTO (N+1)X6                         0745
      SUBROUTINE TRANM(A,B,LL,L)                                               0746
      REAL*8 A,B                                                               0747
      DIMENSION A(LL,6),B(7,L)                                                 0748
      COMMON NUMBER                                                            0749
      NUMBER=NUMBER+1                                                          0750
      A(1,1)=NUMBER                                                            0751
      A(1,2)=B(1,3)                                                            0752
      A(1,3)=B(1,2)                                                            0753
      M1=A(1,2)+1.1                                                            0754
      N=A(1,3)+.1                                                              0755
      DO1019 J=2,M1                                                            0756
      DO1019 K=1,N                                                             0757
      KP=K+1                                                                   0758
      JM=J-1                                                                   0759
      A(J,K)=B(KP,JM)                                                          0760
 1019 CONTINUE                                                                 0761
      RETURN                                                                   0762
      END                                                                      0763
```

```
C   SUBROUTINEULTMA MULTIPLY MATRICES  A(N,M)=B(N,P)XC(PXM),WHERE N,M AND   0764
C   P ARE FROM 6 TO 50                                                     0765
      SUBROUTINE ULTMA(A,B,C,MM,M,L,LL)                                    0766
      REAL*8 A,B,C,T                                                       0767
      DIMENSION A(MM,M),B(MM,L),C(LL,M)                                    0768
      COMMON NUMBER                                                        0769
      NUMBER=NUMBER+1                                                      0770
      A(1,1)=NUMBER                                                        0771
      IF(DABS(B(1,3)-C(1,2))-.1)1015,1012,1012                             0772
 1012 WRITE(6,1013) NUMBER                                                 0773
 1013 FORMAT(48H  NONCCMFORMABLE MATRIX MULTIDEE TO FORM NUMBER , I7)      0774
 3333 STOP                                                                 0775
 1015 A(1,2)=B(1,2)                                                        0776
      A(1,3)=C(1,3)                                                        0777
      M1=A(1,2)+1.1D0                                                      0778
      N=A(1,3)+.1D0                                                        0779
      NN=B(1,3)+.1D0                                                       0780
      DO 1017 I=2,M1                                                       0781
      DO 1017 J=1,N                                                        0782
      T=0.D0                                                               0783
      DO 1018 K=1,NN                                                       0784
      KK=K+1                                                               0785
      T=T+B(I,K)*C(KK,J)                                                   0786
 1018 CONTINUE                                                             0787
      A(I,J)=T                                                             0788
 1017 CONTINUE                                                             0789
      RETURN                                                               0790
      END                                                                  0791
C   SUBROUTINE ULTMB MULTIPLY MATRIX BY COLUMN                             0792
      SUBROUTINE ULTMB(A,B,C,LL,L)                                         0793
      REAL*8 A,B,C,T                                                       0794
      DIMENSION A(LL),B(LL,L),C(LL)                                        0795
      COMMON NUMBER                                                        0796
      NUMBER=NUMBER+1                                                      0797
      IF(DABS(B(1,3)-C(1))-.1D0)25,26,26                                   0798
   26 WRITE(6,27) NUMBER                                                   0799
   27 FORMAT(38H  NONCCMFORMABLE ULTMB TO GET NUMBER ,I7)                  0800
      STOP                                                                 0801
   25 A(1)=B(1,2)                                                          0802
      M=B(1,2)+1.1D0                                                       0803
      N=B(1,3)+.1D0                                                        0804
      DO 28 I=2,M                                                          0805
      T=0.D0                                                               0806
      DO 29 J=1,N                                                          0807
      JJ=J+1                                                               0808
      T=T+B(I,J)*C(JJ)                                                     0809
   29 CONTINUE                                                             0810
      A(I)=T                                                               0811
   28 CONTINUE                                                             0812
      RETURN                                                               0813
      END                                                                  0814
```

```
      SUBROUTINE OUTPUT(R,B,U,V,T1,T2,T3,T5,LL,L,BETA)                  20000
      IMPLICIT REAL*8(A-H,O-Z),INTEGER(I-N)                             20010
      DIMENSION R(7,6),B(7,6),U(7,L),V(7,L),T1(7,L),T2(7,L),T3(LL,6),   20020
     1T4(7,6),O(5),T5(LL,6),D(7,6),E(7,6)                               20030
      WRITE(6,32)                                                       20040
   32 FORMAT(19HO DESIRED MATRIX IS  )                                  20050
      CALL PRINM(R,6)                                                   20060
      CALL ULTMA(T1,R,U,7,L,6,7)                                        20070
      CALL ADDEM(T2,T1,V,-1.D0,7,L)                                     20080
      CALL TRANM(T3,T2,LL,L)                                            20090
      CALL ULTMA(T4,T2,T3,7,6,L,LL)                                     20100
      CALL TRACM(A,T4)                                                  20110
      O(1)=A                                                            20120
      CALL ULTMA(T2,B,V,7,L,6,7)                                        20130
      CALL ADDEM(T1,U,T2,-1.D0,7,L)                                     20140
      CALL TRANM(T5,T1,LL,L)                                            20150
      CALL ULTMA(D,T1,T5,7,6,L,LL)                                      20160
      CALL TRACM(A,D)                                                   20170
      O(2)=A                                                            20180
      CALL ULTMA(E,T1,T3,7,6,L,LL)                                      20190
      CALL TRACM(A,E)                                                   20200
      O(3)=A                                                            20210
      CALL ADDEM(E,T4,D,BETA,7,6)                                       20220
      TETA=1.D0+BETA                                                    20230
      DO 35 I=2,7                                                       20240
      DO 35 J=1,6                                                       20250
      E(I,J)=E(I,J)/TETA                                                20260
   35 CONTINUE                                                          20270
      CALL TRACM(A,E)                                                   20280
      O(4)=A                                                            20290
      WRITE(6,40)BETA                                                   20300
   40 FORMAT(9HO  BETA = , D12.6 )                                      20310
      DO 30 I=1,4                                                       20320
   30 WRITE(6,31)I,O(I)                                                 20330
   31 FORMAT(20HO  CRITERION NUMBER ,I4,8H EQUALS ,1PE14.6)             20340
      RETURN                                                            20350
      END                                                               20360
```

APPENDIX C.

SAMPLE PROBLEM SOLVED BY COMPUTER

C-1 Sample Problem Using Program CECI 1 with 0% and 10% Maximum Random Errors.

The nomenclature used in the output of the program is as follows:

MODE = 1	Data will be generated in the program.
Matrix P	Original matrix C as read from cards.
Matrix BE	Matrix U as read from cards.
Matrix EB	Matrix V from V = CU.
Matrix U	Matrix BE with random errors introduced.
Matrix V	Matrix EB with random errors introduced.
Output No. 1	The desired compliance matrix which satisfies criterion one.
Output No. 2	The desired compliance matrix which satisfies criterion two.
Output No. 3A	The desired compliance matrix which satisfies criterion three, using the iterative procedure for taking the square root of the non-symmetric matrix VV^TUU^T.
Output No. 3B	The desired compliance matrix which satisfies criterion three, using the iterative procedure of taking the square root of the non-symmetric matrix UU^TVV^T.
Output No. 3C	The desired compliance matrix which satisfies criterion three. It is the average of outputs 3A and 3B.
Output No. 3D	The desired compliance matrix which satisfies criterion three, using the procedure that twice takes square roots of symmetric matrices.

The printed output for the problem with zero error appears on pages 77 to 82 inclusive; that for the problem with 10% maximum error appears on pages 83 to 88 inclusive.

MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

0.9000D 01	0.0	0.0	0.0	0.0	0.0
0.0	0.1800D 02	0.0	0.0	0.0	0.0
0.0	0.0	0.5000D 01	0.0	0.0	0.0
0.0	0.0	0.0	0.1200D 02	0.0	0.0
0.0	0.0	0.0	0.0	0.8000D 01	0.0
0.0	0.0	0.0	0.0	0.0	0.2700D 02

MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS,24 COLUMNS

-0.9700D 02	0.9200D 02	-0.1900D 02	0.3800D 02	-0.9900D 02	0.2430D 02	0.7000D 00	-0.5020D 02
0.3200D 02	-0.8840D 02	0.0	0.5100D 02	0.7500D 02	0.1200D 02	-0.2800D 02	0.6240D 02
-0.5000D 00	0.5160D 02	-0.7160D 02	0.4300D 01	0.1460D 02	0.2350D 02	-0.3670D 02	0.4500D 02
0.9000D 01	0.4500D 02	-0.2000D 02	0.8700D 02	0.9700D 02	0.1050D 02	0.2080D 02	-0.5210D 02
0.9870D 02	-0.2690D 02	0.9500D 02	0.1300D 02	-0.7900D 02	0.6700D 02	0.1700D 02	0.2500D 01
-0.5470D 02	0.3590D 02	-0.6550D 02	0.5550D 02	-0.5000D 00	0.7510D 02	0.3450D 02	0.8210D 02
0.4200D 02	-0.2700D 02	0.3400D 02	0.1900D 02	0.6200D 02	0.3500D 02	0.0	0.8700D 02
-0.5100D 02	0.3500D 02	0.1100D 02	-0.5300D 02	0.2400D 02	0.6000D 02	0.1500D 02	-0.3200D 02
-0.6900D 02	0.4200D 02	0.3100D 02	0.1500D 02	0.9810D 02	-0.4600D 01	0.3200D 01	0.6630D 02
0.1400D 02	0.3400D 02	-0.1500D 02	0.0	-0.8500D 02	0.5100D 02	-0.3900D 02	0.7800D 02
0.6400D 02	0.4100D 02	0.4100D 02	-0.2100D 02	-0.2300D 02	0.3600D 02	0.6100D 02	0.3100D 02
-0.1600D 02	-0.2000D 01	0.6700D 02	-0.5800D 02	0.2150D 02	-0.5500D 02	0.6580D 02	0.7590D 02
0.2500D 02	-0.5600D 02	0.6900D 02	0.3200D 02	0.5000D 01	0.0	-0.1300D 02	0.8300D 02
0.2000D 01	-0.3900D 02	-0.1300D 02	0.5100D 02	0.1100D 02	-0.1500D 02	-0.2100D 02	-0.2500D 01
0.4420D 02	-0.8000D 00	0.5500D 02	0.3350D 02	0.5300D 01	-0.8600D 01	0.6310D 02	0.1480D 02
0.5630D 02	0.3200D 02	-0.2260D 02	0.4550D 02	-0.5560D 02	-0.3500D 02	0.8900D 02	0.4800D 02
-0.6700D 02	0.2100D 02	-0.5000D 01	-0.8000D 01	0.4700D 02	0.1500D 02	0.2900D 02	-0.2200D 02
0.4900D 02	-0.2900D 02	0.6400D 02	0.7600D 02	0.5690D 02	-0.3780D 02	0.3820D 02	-0.1950D 02

MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS,24 COLUMNS

```
-0.8730D 03  0.8280D 03 -0.1710D 03  0.3420D 03 -0.8910D 03  0.2187D 03  0.6300D 01 -0.4518D 03
 0.2880D 03 -0.7956D 03  0.0         0.4590D 03  0.6750D 03  0.1080D 03 -0.2520D 03  0.5616D 03
-0.4500D 01  0.4644D 03 -0.6444D 03  0.3870D 02  0.1314D 03  0.2115D 03 -0.3303D 03  0.4050D 03
 0.1620D 03  0.8100D 03 -0.3600D 03  0.1566D 04  0.1746D 04  0.1890D 03  0.3744D 03 -0.9378D 03
 0.1777D 04 -0.4842D 03  0.1710D 04  0.2340D 03 -0.1422D 04  0.1206D 04  0.3060D 03  0.4500D 02
-0.9846D 03  0.6462D 03 -0.1179D 04  0.9990D 03 -0.9000D 01  0.1352D 04  0.6210D 03  0.1478D 04
 0.2100D 03 -0.1350D 03  0.1700D 03  0.9500D 02  0.3100D 03  0.1750D 03  0.0         0.4350D 03
-0.2550D 03  0.1750D 03  0.5500D 02 -0.2650D 03  0.1200D 03  0.3000D 03  0.7500D 02 -0.1600D 03
-0.3450D 03  0.2100D 03  0.1550D 03  0.7500D 02  0.4905D 03 -0.2300D 02  0.1600D 02  0.3315D 03
 0.1680D 03  0.4080D 03 -0.1800D 03  0.0        -0.1020D 04  0.6120D 03 -0.4680D 03  0.9360D 03
 0.7680D 03  0.4920D 03  0.4920D 03 -0.2520D 03 -0.2760D 03  0.4320D 03  0.7320D 03  0.3720D 03
-0.1920D 03 -0.2400D 02  0.8040D 03 -0.6960D 03  0.2580D 03 -0.6600D 03  0.7896D 03  0.9108D 03
 0.2000D 03 -0.4480D 03  0.5520D 03  0.2560D 03 -0.4000D 02  0.0        -0.1040D 03  0.6640D 03
 0.1600D 02 -0.3120D 03 -0.1040D 03  0.4080D 03  0.8800D 02 -0.1200D 03 -0.1680D 03 -0.2000D 02
 0.3536D 03 -0.6400D 01  0.4400D 03  0.2680D 03  0.4240D 02 -0.6880D 02  0.5048D 03  0.1184D 03
 0.1520D 04  0.8640D 03 -0.6102D 03  0.1228D 04 -0.1501D 04 -0.9450D 03  0.2403D 04  0.1296D 04
-0.1809D 04  0.5670D 03 -0.1350D 03 -0.2160D 03  0.1269D 04  0.4050D 03  0.7830D 03 -0.5940D 03
 0.1323D 04 -0.7830D 03  0.1728D 04  0.2052D 04  0.1536D 04 -0.1021D 04  0.1031D 04 -0.5265D 03
```

ERROR IN BLANCA = 0.0

ERROR IN BLANCA = 0.0

MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS,24 COLUMNS

```
-0.9700D 02  0.9200D 02 -0.1900D 02  0.3800D 02 -0.9900D 02  0.2430D 02  0.7000D 00 -0.5020D 02
 0.3200D 02 -0.8840D 02  0.0         0.5100D 02  0.7500D 02  0.1200D 02 -0.2800D 02  0.6240D 02
-0.5000D 00  0.5160D 02 -0.7160D 02  0.4300D 01  0.1460D 02  0.2350D 02 -0.3670D 02  0.4500D 02
 0.9000D 01  0.4500D 02 -0.2000D 02  0.8700D 02  0.9700D 02  0.1050D 02  0.2080D 02 -0.5210D 02
 0.9870D 02 -0.2690D 02  0.9500D 02  0.1300D 02 -0.7900D 02  0.6700D 02  0.1700D 02  0.2500D 01
-0.5470D 02  0.3590D 02 -0.6550D 02  0.5550D 02 -0.5000D 00  0.7510D 02  0.3450D 02  0.8210D 02
 0.4200D 02 -0.2700D 02  0.3400D 02  0.1900D 02  0.6200D 02  0.3500D 02  0.0         0.8700D 02
-0.5100D 02  0.3500D 02  0.1100D 02 -0.5300D 02  0.2400D 02  0.6000D 02  0.1500D 02 -0.3200D 02
-0.6900D 02  0.4200D 02  0.3100D 02  0.1500D 02  0.9810D 02 -0.4600D 01  0.3200D 01  0.6630D 02
 0.1400D 02  0.3400D 02 -0.1500D 02  0.0        -0.8500D 02  0.5100D 02 -0.3900D 02  0.7800D 02
 0.6400D 02  0.4100D 02  0.4100D 02 -0.2100D 02 -0.2300D 02  0.3600D 02  0.6100D 02  0.3100D 02
-0.1600D 02 -0.2000D 01  0.6700D 02 -0.5800D 02  0.2150D 02 -0.5500D 02  0.6580D 02  0.7590D 02
 0.2500D 02 -0.5600D 02  0.6900D 02  0.3200D 02  0.5000D 01  0.0        -0.1300D 02  0.8300D 02
 0.2000D 01 -0.3900D 02 -0.1300D 02  0.5100D 02  0.1100D 02 -0.1500D 02 -0.2100D 02 -0.2500D 01
 0.4420D 02 -0.8000D 00  0.5500D 02  0.3350D 02  0.5300D 01 -0.8600D 01  0.6310D 02  0.1480D 02
 0.5630D 02  0.3200D 02 -0.2260D 02  0.4550D 02 -0.5560D 02 -0.3500D 02  0.8900D 02  0.4800D 02
-0.6700D 02  0.2100D 02 -0.5000D 01 -0.8000D 01  0.4700D 02  0.1500D 02  0.2900D 02 -0.2200D 02
 0.4900D 02 -0.2900D 02  0.6400D 02  0.7600D 02  0.5690D 02 -0.3780D 02  0.3820D 02 -0.1950D 02
```

MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS,24 COLUMNS

```
-0.8730D 03  0.8280D 03 -0.1710D 03  0.3420D 03 -0.8910D 03  0.2187D 03  0.6300D 01 -0.4518D 03
 0.2880D 03 -0.7956D 03  0.0         0.4590D 03  0.6750D 03  0.1080D 03 -0.2520D 03  0.5616D 03
-0.4500D 01  0.4644D 03 -0.6444D 03  0.3870D 02  0.1314D 03  0.2115D 03 -0.3303D 03  0.4050D 03
 0.1620D 03  0.8100D 03 -0.3600D 03  0.1566D 04  0.1746D 04  0.1890D 03  0.3744D 03 -0.9378D 03
 0.1777D 04 -0.4842D 03  0.1710D 04  0.2340D 03 -0.1422D 04  0.1206D 04  0.3060D 03  0.4500D 02
-0.9846D 03  0.6462D 03 -0.1179D 04  0.9990D 03 -0.9000D 01  0.1352D 04  0.6210D 03  0.1478D 04
 0.2100D 03 -0.1350D 03  0.1700D 03  0.9500D 02  0.3100D 03  0.1750D 03 -0.0         0.4350D 03
-0.2550D 03  0.1750D 03  0.5500D 02 -0.2650D 03  0.1200D 03  0.3000D 03  0.7500D 02 -0.1600D 03
-0.3450D 03  0.2100D 03  0.1550D 03  0.7500D 02  0.4905D 03 -0.2300D 02  0.1600D 02  0.3315D 03
 0.1680D 03  0.4080D 03 -0.1800D 03  0.0         -0.1020D 04  0.6120D 03 -0.4680D 03  0.9360D 03
 0.7680D 03  0.4920D 03  0.4920D 03 -0.2520D 03 -0.2760D 03  0.4320D 03  0.7320D 03  0.3720D 03
-0.1920D 03 -0.2400D 02  0.8040D 03 -0.6960D 03  0.2580D 03 -0.6600D 03  0.7896D 03  0.9108D 03
 0.2000D 03 -0.4480D 03  0.5520D 03  0.2560D 03  0.4000D 02  0.0         -0.1040D 03  0.6640D 03
 0.1600D 02 -0.3120D 03 -0.1040D 03  0.4080D 03  0.8800D 02 -0.1200D 03 -0.1680D 03 -0.2000D 02
 0.3536D 03 -0.6400D 01  0.4400D 03  0.2680D 03  0.4240D 02 -0.6880D 02  0.5048D 03  0.1184D 03
 0.1520D 04  0.8640D 03 -0.6102D 03  0.1228D 04 -0.1501D 04 -0.9450D 03  0.2403D 04  0.1296D 04
-0.1809D 04  0.5670D 03 -0.1350D 03 -0.2160D 03  0.1269D 04  0.4050D 03  0.7830D 03 -0.5940D 03
 0.1323D 04 -0.7830D 03  0.1728D 04  0.2052D 04  0.1536D 04 -0.1021D 04  0.1031D 04 -0.5265D 03
```

ELAPSED TIME = 10.37

OUTPUT NO. 1,MINIMIZES V-SQUARE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 17, 6 ROWS, 6 COLUMNS

```
 0.9000D 01 -0.1388D-15 -0.7355D-15 -0.8327D-16 -0.3886D-15 -0.2665D-14
-0.1388D-15  0.1800D 02  0.3789D-14  0.1346D-14 -0.9437D-15 -0.1088D-13
-0.7355D-15  0.3789D-14  0.5000D 01 -0.1069D-14  0.3053D-15  0.4885D-14
-0.8327D-16  0.1346D-14 -0.1069D-14  0.1200D 02  0.1998D-14  0.8882D-15
-0.3886D-15 -0.9437D-15  0.3053D-15  0.1998D-14  0.8000D 01  0.1044D-13
-0.2665D-14 -0.1088D-13  0.4885D-14  0.8882D-15  0.1044D-13  0.2700D 02
```

CRITERION NUMBER 1 EQUALS 1.797180D-22

CRITERION NUMBER 2 EQUALS 3.555419D-25

CRITERION NUMBER 3 EQUALS 4.837873D-24

ELAPSED TIME = 1.48

```
 CUTPUT NC. 2, MINIMIZES U-SQUARE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER       33,    6 ROWS, 6 COLUMNS

 C.9000D 01  0.7640D-15 -0.2049D-15  0.1874D-15 -0.1027D-14  0.1910D-14
 0.7640D-15  0.1800D 02 -0.9709D-15 -0.6674D-15  0.4137D-15  0.1265D-13
-0.2049D-15 -0.9709D-15  0.5000D 01  0.5529D-16 -0.1214D-14  0.5123D-16
 0.1874D-15 -0.6674D-15  0.5529D-16  0.1200D 02  0.2082D-15 -0.6147D-15
-0.1027D-14  0.4137D-15 -0.1214D-14  0.2082D-15  0.8000D 01 -0.9368D-15
 0.1910D-14  0.1265D-13  0.5123D-16 -0.6147D-15 -0.9368D-15  0.2700D 02

 CRITERICN NUMBER    1 EQUALS    1.006397D-22

 CRITERICN NUMBER    2 EQUALS    7.975143D-25

 CRITERICN NUMBER    3 EQUALS    6.675377D-24

 ELAPSED TIME =          0.96

 OUTPU NO.3A, MINIMIZES TRIANGLE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER       52,    6 ROWS, 6 COLUMNS

 0.9000D 01  0.4025D-15  0.2234D-14 -0.9714D-15 -0.2665D-14  0.1110D-14
 0.1101D-13  0.1800D 02  0.1732D-13 -0.3331D-15 -0.9548D-14 -0.2198D-13
 0.1908D-15  0.1360D-14  0.5000D 01 -0.3738D-14 -0.1416D-14  0.1110D-14
 0.1527D-14 -0.6384D-15  0.3553D-14  0.1200D 02  0.2442D-14 -0.3109D-14
 0.1069D-14 -0.6661D-15 -0.3553D-14  0.1568D-14  0.8000D 01 -0.6661D-15
 0.0        -0.1155D-13  0.1021D-13 -0.2665D-14  0.1577D-13  0.2700D 02

 CRITERICN NUMBER    1 EQUALS    5.245247D-22

 CRITERICN NUMBER    2 EQUALS    8.446838D-25

 CRITERICN NUMBER    3 EQUALS    1.833622D-23

 ELAPSEC TIME =          0.91
```

```
 OUTPUT NO.3B, MINIMIZES TRIANGLE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER       7C,    6 ROWS, 6 COLUMNS

 0.9000D 01   0.1046D-13   0.3738D-15   0.1707D-14   0.2137D-14   0.8882D-15
 0.8465D-15   0.1800D 02   0.1152D-14  -0.4163D-16  -0.8882D-15  -0.1510D-13
 0.2012D-14   0.1754D-13   0.5000D 01   0.3331D-14  -0.2887D-14   0.7327D-14
-0.3886D-15  -0.1305D-14  -0.3718D-14   0.1200D 02   0.4163D-15  -0.2220D-14
-0.5107D-14  -0.1465D-13  -0.1429D-14   0.5551D-14   0.8000D 01   0.1754D-13
 0.1110D-14  -0.1799D-13   0.1776D-14  -0.3331D-14  -0.4441D-15   0.2700D 02

 CRITERION NUMBER     1 EQUALS    6.412928D-22

 CRITERION NUMBER     2 EQUALS    2.732927D-24

 CRITERION NUMBER     3 EQUALS    3.17C094D-23

 ELAPSED TIME =          0.87

 OUTPUT NO.3C, MINIMIZES TRIANGLE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER       68,    6 ROWS, 6 COLUMNS

 0.9000D 01   0.5433D-14   C.1304D-14   0.3678D-15  -0.2637D-15   0.9992D-15
 0.5926D-14   0.1800D 02   0.9236D-14  -0.1874D-15  -0.5218D-14  -0.1854D-13
 0.1102D-14   0.9451D-14   0.5000D 01  -0.2038D-15  -0.2151D-14   0.4219D-14
 0.5690D-15  -0.9714D-15  -0.8283D-16   0.1200D 02   0.1429D-14  -0.2665D-14
-0.2019D-14  -0.7661D-14  -0.2491D-14   0.3560D-14   0.8000D 01   0.8438D-14
 0.5551D-15  -0.1477D-13   0.5995D-14  -0.2998D-14   0.7661D-14   0.2700D 02

 CRITERION NUMBER     1 EQUALS    5.531635D-22

 CRITERION NUMBER     2 EQUALS    1.355157D-24

 CRITERION NUMBER     3 EQUALS    2.229371D-23
```

```
 OUTPUT NC.3D, MINIMIZES TRIANGLE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER      109,    6 ROWS, 6 COLUMNS

 0.8934D 01   0.1787D 00 -0.1065D 00   0.8814D-01   0.1673D 00 -0.3765D 00
 0.1787D 00   0.1804D 02   0.9723D 00   0.2940D 00 -0.2398D 00   0.4100D-01
-0.1065D 00   0.9723D 00   0.4769D 01   0.8368D 00   0.2083D-01   0.9612D 00
 0.8814D-01   0.2940D 00   0.8368D 00   0.1193D 02   0.4856D 00   0.9662D 00
 0.1673D 00 -0.2398D 00   0.2083D-01   0.4856D 00   0.7768D 01   0.1241D 01
-0.3765D 00   0.4100D-01   0.9612D 00   0.9662D 00   0.1241D 01   0.2681D 02

 CRITERION NUMBER     1 EQUALS    6.446506D 05

 CRITERION NUMBER     2 EQUALS    8.064172D 03

 CRITERION NUMBER     3 EQUALS    5.876357D 04

 ELAPSED TIME =          1.89

 TOTAL ELAPSED TIME =         16.58
```

MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

C.900CD C1	0.C	C.0	0.0	0.0	0.0
0.0	0.1800D C2	0.0	0.C	0.0	0.0
0.0	0.0	0.50C0D 01	0.C	0.0	0.0
0.0	0.0	0.0	0.12C0D 02	0.0	0.0
0.0	0.0	C.C	0.C	0.8000D 01	0.0
0.0	0.0	0.0	0.0	0.0	0.2700D 02

MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS,24 COLUMNS

-0.970CD 02	0.9200D 02	-0.1900D 02	0.3800D C2	-0.9900D 02	0.2430D 02	0.7000D 00	-0.5020D 02
0.3200D 02	-0.8840D 02	0.0	0.5100D 02	0.7500D 02	0.1200D 02	-0.2800D 02	0.6240D 02
-0.500CD 00	0.5160D 02	-C.716CD 02	0.4300D 01	0.1460D 02	0.2350D 02	-0.3670D 02	0.4500D 02
0.9000D 01	0.4500D C2	-C.20C0D 02	0.8700D 02	0.9700D 02	0.1050D 02	0.2080D 02	-0.5210D 02
0.987CD C2	-0.2690D C2	C.950CD 02	0.1300D 02	-0.7900D 02	0.6700D 02	0.1700D 02	0.2500D 01
-0.547CD 02	0.3590D 02	-0.6550D 02	0.5550D 02	-0.5000D 00	0.7510D 02	0.3450D 02	0.8210D 02
0.420CD 02	-0.2700D 02	C.340CD C2	0.1900D 02	0.6200D 02	0.3500D 02	0.0	0.8700D 02
-0.510CD 02	0.3500D C2	C.110CD 02	-0.5300D 02	0.2400D 02	0.6000D 02	0.1500D 02	-0.3200D 02
-0.690CD 02	0.4200D 02	0.310CD 02	0.1500D 02	0.9810D 02	-0.4600D 01	0.3200D 01	0.6630D 02
0.140CD 02	0.3400D C2	-C.1500D 02	0.C	-0.8500D 02	0.5100D 02	-0.3900D 02	0.7800D 02
0.640CD 02	C.4100D C2	C.410CD 02	-0.2100D 02	-0.2300D 02	0.3600D 02	0.6100D C2	0.3100D 02
-0.160CD 02	-C.2000D C1	0.670CD 02	-0.5800D 02	0.2150D 02	-0.5500D 02	0.6580D 02	0.7590D 02
0.2500D 02	-0.5600D 02	0.690CD C2	0.3200D 02	0.5000D 01	0.0	-0.1300D 02	0.8300D 02
0.200CD 01	-0.3900D C2	-C.130CD 02	0.5100D 02	0.1100D 02	-0.1500D 02	-0.2100D 02	-0.2500D 01
0.4420D 02	-0.8000D 00	0.5500D 02	0.3350D 02	0.5300D 01	-0.8600D 01	0.6310D 02	0.1480D 02
0.563CD C2	0.3200D 02	-0.2260D 02	0.4550D C2	-0.5560D 02	-0.3500D 02	0.8900D 02	0.4800D 02
-0.670CD 02	0.2100D 02	-0.500CD 01	-0.8000D 01	0.4700D 02	0.1500D 02	0.2900D 02	-0.2200D 02
0.4900D 02	-0.2900D 02	0.640CD 02	0.7600D 02	0.5690D 02	-0.3780D 02	0.3820D 02	-0.1950D 02

MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS,24 COLUMNS

-0.8730D 03	0.8280D 03	-0.1710D 03	0.3420D 03	-0.8910D 03	0.2187D 03	0.6300D 01	-0.4518D 03
0.2880D 03	-0.7956D 03	0.0	0.4590D 03	0.6750D 03	0.1080D 03	-0.2520D 03	0.5616D 03
-0.4500D 01	0.4644D 03	-0.6444D 03	0.3870D 02	0.1314D 03	0.2115D 03	-0.3303D 03	0.4050D 03
0.1620D 03	0.8100D 03	-0.3600D 03	0.1566D 04	0.1746D 04	0.1890D 03	0.3744D 03	-0.9378D 03
0.1777D 04	-0.4842D 03	0.1710D 04	0.2340D 03	-0.1422D 04	0.1206D 04	0.3060D 03	0.4500D 02
-0.9846D 03	0.6462D 03	-0.1179D 04	0.9990D 03	-0.9000D 01	0.1352D 04	0.6210D 03	0.1478D 04
0.2100D 03	-0.1350D 03	0.1700D 03	0.9500D 02	0.3100D 03	0.1750D 03	0.0	0.4350D 03
-0.2550D 03	0.1750D 03	0.5500D 02	-0.2650D 03	0.1200D 03	0.3000D 03	0.7500D 02	-0.1600D 03
-0.3450D 03	0.2100D 03	0.1550D 03	0.7500D 02	0.4905D 03	-0.2300D 02	0.1600D 02	0.3315D 03
0.1680D 03	0.4080D 03	-0.1800D 03	0.0	-0.1020D 04	0.6120D 03	-0.4680D 03	0.9360D 03
0.7680D 03	0.4920D 03	0.4920D 03	-0.2520D 03	-0.2760D 03	0.4320D 03	0.7320D 03	0.3720D 03
-0.1920D 03	-0.2400D 02	0.8040D 03	-0.6960D 03	0.2580D 03	-0.6600D 03	0.7896D 03	0.9108D 03
0.2000D 03	-0.4480D 03	0.5520D 03	0.2560D 03	0.4000D 02	0.0	-0.1040D 03	0.6640D 03
0.1600D 02	-0.3120D 03	-0.1040D 03	0.4080D 03	0.8800D 02	-0.1200D 03	-0.1680D 03	-0.2000D 02
0.3536D 03	-0.6400D 01	0.4400D 03	0.2680D 03	0.4240D 02	-0.6880D 02	0.5048D 03	0.1184D 03
0.1520D 04	0.8640D 03	-0.6102D 03	0.1228D 04	-0.1501D 04	-0.9450D 03	0.2403D 04	0.1296D 04
-0.1809D 04	0.5670D 03	-0.1350D 03	-0.2160D 03	0.1269D 04	0.4050D 03	0.7830D 03	-0.5940D 03
0.1323D 04	-0.7830D 03	0.1728D 04	0.2052D 04	0.1536D 04	-0.1021D 04	0.1031D 04	-0.5265D 03

ERROR IN BLANCA = 0.01000

ERROR IN BLANCA = 0.01000

MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS,24 COLUMNS

-0.1052D 03	0.8658D 02	-0.2068D 02	0.4062D 02	-0.9314D 02	0.2531D 02	0.7619D 00	-0.5323D 02
0.2964D 02	-0.9087D 02	0.2318D 00	0.4927D 02	0.6915D 02	0.1111D 02	-0.2937D 02	0.5892D 02
-0.5106D 00	0.5199D 02	-0.7779D 02	0.3884D 01	0.1384D 02	0.2411D 02	-0.3850D 02	0.4176D 02
0.8827D 01	0.4064D 02	-0.2122D 02	0.9073D 02	0.8744D 02	0.1096D 02	0.2265D 02	-0.5227D 02
0.9674D 02	-0.2810D 02	0.1029D 03	0.1231D 02	-0.8549D 02	0.7140D 02	0.1641D 02	0.2675D 01
-0.5910D 02	0.3826D 02	-0.6028D 02	0.5143D 02	-0.4528D 00	0.7832D 02	0.3386D 02	0.7743D 02
0.3815D 02	-0.2780D 02	0.3433D 02	0.1980D 02	0.6712D 02	0.3157D 02	0.2069D 00	0.9459D 02
-0.5325D 02	0.3524D 02	0.9924D 01	-0.5662D 02	0.2505D 02	0.6025D 02	0.1386D 02	-0.3158D 02
-0.6931D 02	0.4202D 02	0.3309D 02	0.1365D 02	0.9464D 02	-0.4179D 01	0.2968D 01	0.6291D 02
0.1522D 02	0.3199D 02	-0.1602D 02	0.5648D-01	-0.9211D 02	0.5216D 02	-0.4008D 02	0.7667D 02
0.6430D 02	0.4127D 02	0.3946D 02	-0.1934D 02	-0.2357D 02	0.3465D 02	0.6634D 02	0.3124D 02
-0.1679D 02	-0.2094D 01	0.6757D 02	-0.6069D 02	0.1946D 02	-0.4966D 02	0.7113D 02	0.7766D 02
0.2273D 02	-0.5197D 02	0.6779D 02	0.3337D 02	0.5115D 01	0.4043D 00	-0.1226D 02	0.8964D 02
0.1853D 01	-0.3511D 02	-0.1389D 02	0.4827D 02	0.1037D 02	-0.1360D 02	-0.2278D 02	-0.2450D 01
0.4549D 02	-0.8394D 00	0.5651D 02	0.3226D 02	0.5741D 01	-0.9329D 01	0.6715D 02	0.1427D 02
0.5346D 02	0.3151D 02	-0.2095D 02	0.4126D 02	-0.5709D 02	-0.3303D 02	0.8546D 02	0.4351D 02
-0.6054D 02	0.2076D 02	-0.4913D 01	-0.8333D 01	0.4795D 02	0.1575D 02	0.2921D 02	-0.2251D 02
0.4815D 02	-0.2964D 02	0.6181D 02	0.7452D 02	0.6197D 02	-0.4012D 02	0.3980D 02	-0.2071D 02

```
 CUTPUT NO. 2, MINIMIZES U-SQUARE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER       33,    6 ROWS, 6 COLUMNS

 0.9357D 01 -0.1664D 00 -0.3592D-01  0.3113D 00  0.1078D 00  0.2026D 00
-0.1664D 00  0.1811D 02  0.8948D-01 -0.1677D 00  0.1174D 00 -0.1237D 00
-0.3592D-01  0.8948D-01  0.4771D 01  0.1212D 00 -0.4547D-01  0.5898D-01
 0.3113D 00 -0.1677D 00  0.1212D 00  0.1178D 02 -0.2360D 00 -0.5429D-01
 0.1078D 00  0.1174D 00 -0.4547D-01 -0.2360D 00  0.7627D 01 -0.4839D-01
 0.2026D 00 -0.1237D 00  0.5898D-01 -0.5429D-01 -0.4839D-01  0.2701D 02

 CRITERION NUMBER     1 EQUALS    5.422693D 05

 CRITERION NUMBER     2 EQUALS    2.133526D 03

 CRITERION NUMBER     3 EQUALS    3.067654D 04

 ELAPSED TIME =          1.65

 CUTPU NO.3A, MINIMIZES TRIANGLE ERROR
DESIRED MATRIX IS
  MATRIX NUMBER      124,    6 ROWS, 6 COLUMNS

 0.9312D 01 -0.1634D 00 -0.3204D-01  0.3155D 00  0.1280D 00  0.2673D-01
-0.1634D 00  0.1802D 02  0.1128D 00 -0.2446D 00  0.1366D 00 -0.1234D 00
-0.3204D-01  0.1128D 00  0.4725D 01  0.1868D 00 -0.6587D-01  0.1538D 00
 0.3155D 00 -0.2446D 00  0.1868D 00  0.1174D 02 -0.2198D 00 -0.1347D 00
 0.1280D 00  0.1366D 00 -0.6587D-01 -0.2198D 00  0.7596D 01 -0.7312D-02
 0.2673D-01 -0.1234D 00  0.1538D 00 -0.1347D 00 -0.7312D-02  0.2688D 02

 CRITERION NUMBER     1 EQUALS    5.145427D 05

 CRITERION NUMBER     2 EQUALS    2.193255D 03

 CRITERION NUMBER     3 EQUALS    3.012705D 04

 ELAPSED TIME =          1.61
```

MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS,24 COLUMNS

```
-0.9166D 03  0.8613D 03 -0.1583D 03  0.3649D 03 -0.9509D 03  0.2331D 03  0.6178D 01 -0.4439D 03
 0.2950D 03 -0.8119D 03  0.8306D 00  0.4871D 03  0.6817D 03  0.1047D 03 -0.2535D 03  0.5768D 03
-0.4334D 01  0.4954D 03 -0.6116D 03  0.4212D 02  0.1372D 03  0.2082D 03 -0.3194D 03  0.3846D 03
 0.1599D 03  0.7295D 03 -0.3537D 03  0.1674D 04  0.1830D 04  0.1716D 03  0.3467D 03 -0.9999D 03
 0.1850D 04 -0.4361D 03  0.1644D 04  0.2109D 03 -0.1337D 04  0.1117D 04  0.3021D 03  0.4347D 02
-0.1011D 04  0.6908D 03 -0.1109D 04  0.1085D 04 -0.8327D 01  0.1329D 04  0.5754D 03  0.1521D 04
 0.2075D 03 -0.1254D 03  0.1608D 03  0.9320D 02  0.3242D 03  0.1728D 03 -0.9025D 00  0.4184D 03
-0.2565D 03  0.1902D 03  0.4969D 02 -0.2491D 03  0.1205D 03  0.2769D 03  0.8139D 02 -0.1732D 03
-0.3318D 03  0.1937D 03  0.1681D 03  0.7065D 02  0.4526D 03 -0.2274D 02  0.1606D 02  0.3519D 03
 0.1586D 03  0.4164D 03 -0.1665D 03  0.8025D 00 -0.1050D 04  0.5510D 03 -0.4714D 03  0.9405D 03
 0.7717D 03  0.5217D 03  0.4755D 03 -0.2292D 03 -0.2835D 03  0.4693D 03  0.6909D 03  0.3500D 03
-0.1965D 03 -0.2517D 02  0.7592D 03 -0.7166D 03  0.2739D 03 -0.6518D 03  0.7181D 03  0.9916D 03
 0.2135D 03 -0.4431D 03  0.4986D 03  0.2510D 03  0.4003D 02  0.4526D 00 -0.1042D 03  0.6023D 03
 0.1732D 02 -0.3389D 03 -0.1068D 03  0.3769D 03  0.8488D 02 -0.1280D 03 -0.1619D 03 -0.2171D 02
 0.3215D 03 -0.6794D 01  0.3982D 03  0.2751D 03  0.4600D 02 -0.6473D 02  0.4675D 03  0.1290D 03
 0.1554D 04  0.7781D 03 -0.5619D 03  0.1262D 04 -0.1424D 04 -0.1002D 04  0.2274D 04  0.1223D 04
-0.1657D 04  0.5720D 03 -0.1464D 03 -0.2086D 03  0.1147D 04  0.3728D 03  0.7404D 03 -0.6110D 03
 0.1225D 04 -0.8190D 03  0.1702D 04  0.1861D 04  0.1672D 04 -0.1107D 04  0.9753D 03 -0.4759D 03
```

ELAPSED TIME = 10.38

OUTPUT NO. 1,MINIMIZES V-SQUARE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 17, 6 ROWS, 6 COLUMNS

```
 0.9263D 01 -0.1383D 00 -0.1683D-01  0.3284D 00  0.1558D 00 -0.1802D 00
-0.1383D 00  0.1793D 02  0.1647D 00 -0.3364D 00  0.1647D 00 -0.1294D 00
-0.1683D-01  0.1647D 00  0.4647D 01  0.2943D 00 -0.9470D-01  0.3763D 00
 0.3284D 00 -0.3364D 00  0.2943D 00  0.1170D 02 -0.2027D 00 -0.2347D 00
 0.1558D 00  0.1647D 00 -0.9470D-01 -0.2027D 00  0.7574D 01  0.3379D-01
-0.1802D 00 -0.1294D 00  0.3763D 00 -0.2347D 00  0.3379D-01  0.2671D 02
```

CRITERION NUMBER 1 EQUALS 4.998192D 05

CRITERION NUMBER 2 EQUALS 2.584620D 03

CRITERION NUMBER 3 EQUALS 3.145058D 04

ELAPSED TIME = 1.47

OUTPUT NO.3B, MINIMIZES TRIANGLE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 214, 6 ROWS, 6 COLUMNS

```
 0.9312D 01 -0.1634D 00 -0.3204D-01  0.3155D 00  0.1280D 00  0.2673D-01
-0.1634D 00  0.1802D 02  0.1128D 00 -0.2446D 00  0.1366D 00 -0.1234D 00
-0.3204D-01  0.1128D 00  0.4725D 01  0.1868D 00 -0.6587D-01  0.1538D 00
 0.3155D 00 -0.2446D 00  0.1868D 00  0.1174D 02 -0.2198D 00 -0.1347D 00
 0.1280D 00  0.1366D 00 -0.6587D-01 -0.2198D 00  0.7596D 01 -0.7312D-02
 0.2673D-01 -0.1234D 00  0.1538D 00 -0.1347D 00 -0.7312D-02  0.2688D 02
```

CRITERION NUMBER 1 EQUALS 5.145427D 05

CRITERION NUMBER 2 EQUALS 2.193255D 03

CRITERION NUMBER 3 EQUALS 3.012705D 04

ELAPSED TIME = 0.85

OUTPUT NO.3C, MINIMIZES TRIANGLE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 212, 6 ROWS, 6 COLUMNS

```
 0.9312D 01 -0.1634D 00 -0.3204D-01  0.3155D 00  0.1280D 00  0.2673D-01
-0.1634D 00  0.1802D 02  0.1128D 00 -0.2446D 00  0.1366D 00 -0.1234D 00
-0.3204D-01  0.1128D 00  0.4725D 01  0.1868D 00 -0.6587D-01  0.1538D 00
 0.3155D 00 -0.2446D 00  0.1868D 00  0.1174D 02 -0.2198D 00 -0.1347D 00
 0.1280D 00  0.1366D 00 -0.6587D-01 -0.2198D 00  0.7596D 01 -0.7312D-02
 0.2673D-01 -0.1234D 00  0.1538D 00 -0.1347D 00 -0.7312D-02  0.2688D 02
```

CRITERION NUMBER 1 EQUALS 5.145427D 05

CRITERION NUMBER 2 EQUALS 2.193255D 03

CRITERION NUMBER 3 EQUALS 3.012705D 04

OUTPUT NO.3D, MINIMIZES TRIANGLE ERROR

DESIRED MATRIX IS

MATRIX NUMBER 253, 6 ROWS, 6 COLUMNS

```
 0.9312D 01 -0.1634D 00 -0.3204D-01  0.3155D 00  0.1280D 00  0.2673D-01
-0.1634D 00  0.1802D 02  0.1128D 00 -0.2446D 00  0.1366D 00 -0.1234D 00
-0.3204D-01  0.1128D 00  0.4725D 01  0.1868D 00 -0.6587D-01  0.1538D 00
 0.3155D 00 -0.2446D 00  0.1868D 00  0.1174D 02 -0.2198D 00 -0.1347D 00
 0.1280D 00  0.1366D 00 -0.6587D-01 -0.2198D 00  0.7596D 01 -0.7312D-02
 0.2673D-01 -0.1234D 00  0.1538D 00 -0.1347D 00 -0.7312D-02  0.2688D 02
```

CRITERION NUMBER 1 EQUALS 5.145427D 05

CRITERION NUMBER 2 EQUALS 2.193255D 03

CRITERION NUMBER 3 EQUALS 3.012705D 04

ELAPSED TIME = 1.99

TOTAL ELAPSED TIME = 18.06

C-2 Sample Problem Using Program CECI 2 with 0% and 10% Maximum Random Errors.

The nomenclature used in the output of the program is as follows:

MODE=1	Data will be generated in the program
BETA 0.8	(The sequence used is ρ = .1,.2,...,.8.)
Matrix P	Original matrix C as read from cards.
Matrix BE	Matrix U as read from cards.
Matrix EB	Matrix V from V = CU.
Matrix U	Matrix BE with random errors introduced.
Matrix V	Matrix EB with random errors introduced.
Output No. 4	The desired compliance matrix which satisfies criterion four.

The printed output for the problem with zero error appears on pages 90 to 93 inclusive; that for the problem with 10% maximum error appears on pages 94 to 97 inclusive.

MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

0.9000D 01	0.0	0.0	0.0	0.0	0.0
0.0	0.1800D 02	0.0	0.0	0.0	0.0
0.0	0.0	0.5000D 01	0.0	0.0	0.0
0.0	0.0	0.0	0.1200D 02	0.0	0.0
0.0	0.0	0.0	0.0	0.8000D 01	0.0
0.0	0.0	0.0	0.0	0.0	0.2700D 02

MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS,24 COLUMNS

-0.9700D 02	0.9200D 02	-0.1900D 02	0.3800D 02	-0.9900D 02	0.2430D 02	0.7000D 00	-0.5020D 02
0.3200D 02	-0.8840D 02	0.0	0.5100D 02	0.7500D 02	0.1200D 02	-0.2800D 02	0.6240D 02
-0.5000D 00	0.5160D 02	-0.7160D 02	0.4300D 01	0.1460D 02	0.2350D 02	-0.3670D 02	0.4500D 02
0.9000D 01	0.4500D 02	-0.2000D 02	0.8700D 02	0.9700D 02	0.1050D 02	0.2080D 02	-0.5210D 02
0.9870D 02	-0.2690D 02	0.9500D 02	0.1300D 02	-0.7900D 02	0.6700D 02	0.1700D 02	0.2500D 01
-0.5470D 02	0.3590D 02	-0.6550D 02	0.5550D 02	-0.5000D 00	0.7510D 02	0.3450D 02	0.8210D 02
0.4200D 02	-0.2700D 02	0.3400D 02	0.1900D 02	0.6200D 02	0.3500D 02	0.0	0.8700D 02
-0.5100D 02	0.3500D 02	0.1100D 02	-0.5300D 02	0.2400D 02	0.6000D 02	0.1500D 02	-0.3200D 02
-0.6900D 02	0.4200D 02	0.3100D 02	0.1500D 02	0.9810D 02	-0.4600D 01	0.3200D 01	0.6630D 02
0.1400D 02	0.3400D 02	-0.1500D 02	0.0	-0.8500D 02	0.5100D 02	-0.3900D 02	0.7800D 02
0.6400D 02	0.4100D 02	0.4100D 02	-0.2100D 02	-0.2300D 02	0.3600D 02	0.6100D 02	0.3100D 02
-0.1600D 02	-0.2000D 01	0.6700D 02	-0.5800D 02	0.2150D 02	-0.5500D 02	0.6580D 02	0.7590D 02
0.2500D 02	-0.5600D 02	0.6900D 02	0.3200D 02	0.5000D 01	0.0	-0.1300D 02	0.8300D 02
0.2000D 01	-0.3900D 02	-0.1300D 02	0.5100D 02	0.1100D 02	-0.1500D 02	-0.2100D 02	-0.2500D 01
0.4420D 02	-0.8000D 00	0.5500D 02	0.3350D 02	0.5300D 01	-0.8600D 01	0.6310D 02	0.1480D 02
0.5630D 02	0.3200D 02	-0.2260D 02	0.4550D 02	-0.5560D 02	-0.3500D 02	0.8900D 02	0.4800D 02
-0.6700D 02	0.2100D 02	-0.5000D 01	-0.8000D 01	0.4700D 02	0.1500D 02	0.2900D 02	-0.2200D 02
0.4900D 02	-0.2900D 02	0.6400D 02	0.7600D 02	0.5690D 02	-0.3780D 02	0.3820D 02	-0.1950D 02

MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS, 24 COLUMNS

-0.8730D 03	0.8280D 03	-0.1710D 03	0.3420D 03	-0.8910D 03	0.2187D 03	0.6300D 01	-0.4518D 03
0.2880D 03	-0.7956D 03	0.0	0.4590D 03	0.6750D 03	0.1080D 03	-0.2520D 03	0.5616D 03
-0.4500D 01	0.4644D 03	-0.6444D 03	0.3870D 02	0.1314D 03	0.2115D 03	-0.3303D 03	0.4050D 03
0.1620D 03	0.8100D 03	-0.3600D 03	0.1566D 04	0.1746D 04	0.1890D 03	0.3744D 03	-0.9378D 03
0.1777D 04	-0.4842D 03	0.1710D 04	0.2340D 03	-0.1422D 04	0.1206D 04	0.3060D 03	0.4500D 02
-0.9846D 03	0.6462D 03	-0.1179D 04	0.9990D 03	-0.9000D 01	0.1352D 04	0.6210D 03	0.1478D 04
0.2100D 03	-0.1350D 03	0.1700D 03	0.9500D 02	0.3100D 03	0.1750D 03	0.0	0.4350D 03
-0.2550D 03	0.1750D 03	0.5500D 02	-0.2650D 03	0.1200D 03	0.3000D 03	0.7500D 02	-0.1600D 03
-0.3450D 03	0.2100D 03	0.1550D 03	0.7500D 02	0.4905D 03	-0.2300D 02	0.1600D 02	0.3315D 03
0.1680D 03	0.4080D 03	-0.1800D 03	0.0	-0.1020D 04	0.6120D 03	-0.4680D 03	0.9360D 03
0.7680D 03	0.4920D 03	0.4920D 03	-0.2520D 03	-0.2760D 03	0.4320D 03	0.7320D 03	0.3720D 03
-0.1920D 03	-0.2400D 02	0.8040D 03	-0.6960D 03	0.2580D 03	-0.6600D 03	0.7896D 03	0.9108D 03
0.2000D 03	-0.4480D 03	0.5520D 03	0.2560D 03	0.4000D 02	0.0	-0.1040D 03	0.6640D 03
0.1600D 02	-0.3120D 03	-0.1040D 03	0.4080D 03	0.8800D 02	-0.1200D 03	-0.1680D 03	-0.2000D 02
0.3536D 03	-0.6400D 01	0.4400D 03	0.2680D 03	0.4240D 02	-0.6880D 02	0.5048D 03	0.1184D 03
0.1520D 04	0.8640D 03	-0.6102D 03	0.1228D 04	-0.1501D 04	-0.9450D 03	0.2403D 04	0.1296D 04
-0.1809D 04	0.5670D 03	-0.1350D 03	-0.2160D 03	0.1269D 04	0.4050D 03	0.7830D 03	-0.5940D 03
0.1323D 04	-0.7830D 03	0.1728D 04	0.2052D 04	0.1536D 04	-0.1021D 04	0.1031D 04	-0.5265D 03

ERROR IN BLANCA = 0.0

ERROR IN BLANCA = 0.0

MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS, 24 COLUMNS

-0.9700D 02	0.9200D 02	-0.1900D 02	0.3800D 02	-0.9900D 02	0.2430D 02	0.7000D 00	-0.5020D 02
0.3200D 02	-0.8840D 02	0.0	0.5100D 02	0.7500D 02	0.1200D 02	-0.2800D 02	0.6240D 02
-0.5000D 00	0.5160D 02	-0.7160D 02	0.4300D 01	0.1460D 02	0.2350D 02	-0.3670D 02	0.4500D 02
0.9000D 01	0.4500D 02	-0.2000D 02	0.8700D 02	0.9700D 02	0.1050D 02	0.2080D 02	-0.5210D 02
0.9870D 02	-0.2690D 02	0.9500D 02	0.1300D 02	-0.7900D 02	0.6700D 02	0.1700D 02	0.2500D 01
-0.5470D 02	0.3590D 02	-0.6550D 02	0.5550D 02	-0.5000D 00	0.7510D 02	0.3450D 02	0.8210D 02
0.4200D 02	-0.2700D 02	0.3400D 02	0.1900D 02	0.6200D 02	0.3500D 02	0.0	0.8700D 02
-0.5100D 02	0.3500D 02	0.1100D 02	-0.5300D 02	0.2400D 02	0.6000D 02	0.1500D 02	-0.3200D 02
-0.6900D 02	0.4200D 02	0.3100D 02	0.1500D 02	0.9810D 02	-0.4600D 01	0.3200D 01	0.6630D 02
0.1400D 02	0.3400D 02	-0.1500D 02	0.0	-0.8500D 02	0.5100D 02	-0.3900D 02	0.7800D 02
0.6400D 02	0.4100D 02	0.4100D 02	-0.2100D 02	-0.2300D 02	0.3600D 02	0.6100D 02	0.3100D 02
-0.1600D 02	-0.2000D 01	0.6700D 02	-0.5800D 02	0.2150D 02	-0.5500D 02	0.6580D 02	0.7590D 02
0.2500D 02	-0.5600D 02	0.6900D 02	0.3200D 02	0.5000D 01	0.0	-0.1300D 02	0.8300D 02
0.2000D 01	-0.3900D 02	-0.1300D 02	0.5100D 02	0.1100D 02	-0.1500D 02	-0.2100D 02	-0.2500D 01
0.4420D 02	-0.8000D 00	0.5500D 02	0.3350D 02	0.5300D 01	-0.8600D 01	0.6310D 02	0.1480D 02
0.5630D 02	0.3200D 02	-0.2260D 02	0.4550D 02	-0.5560D 02	-0.3500D 02	0.8900D 02	0.4800D 02
-0.6700D 02	0.2100D 02	-0.5000D 01	-0.8000D 01	0.4700D 02	0.1500D 02	0.2900D 02	-0.2200D 02
0.4900D 02	-0.2900D 02	0.6400D 02	0.7600D 02	0.5690D 02	-0.3780D 02	0.3820D 02	-0.1950D 02

MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS,24 COLUMNS

-0.8730D 03	0.8280D 03	-0.1710D 03	0.3420D 03	-0.8910D 03	0.2187D 03	0.6300D 01	-0.4518D 03				
0.2880D 03	-0.7956D 03	0.0	0.4590D 03	0.6750D 03	0.1080D 03	-0.2520D 03	0.5616D 03				
-0.4500D 01	0.4644D 03	-0.6444D 03	0.3870D 02	0.1314D 03	0.2115D 03	-0.3303D 03	0.4050D 03				
0.1620D 03	0.8100D 03	-0.3600D 03	0.1566D 04	0.1746D 04	0.1890D 03	0.3744D 03	-0.9378D 03				
0.1777D 04	-0.4842D 03	0.1710D 04	0.2340D 03	-0.1422D 04	0.1206D 04	0.3060D 03	0.4500D 02				
-0.9846D 03	0.6462D 03	-0.1179D 04	0.9990D 03	-0.9000D 01	0.1352D 04	0.6210D 03	0.1478D 04				
0.2100D 03	-0.1350D 03	0.1700D 03	0.9500D 02	0.3100D 03	0.1750D 03	-0.0	0.4350D 03				
-0.2550D 03	0.1750D 03	0.5500D 02	-0.2650D 03	0.1200D 03	0.3000D 03	0.7500D 02	-0.1600D 03				
-0.3450D 03	0.2100D 03	0.1550D 03	0.7500D 02	0.4905D 03	-0.2300D 02	0.1600D 02	0.3315D 03				
0.1680D 03	0.4080D 03	-0.1800D 03	0.0	-0.1020D 04	0.6120D 03	-0.4680D 03	0.9360D 03				
0.7680D 03	0.4920D 03	0.4920D 03	-0.2520D 03	-0.2760D 03	0.4320D 03	0.7320D 03	0.3720D 03				
-0.1920D 03	-0.2400D 02	0.8040D 03	-0.6960D 03	0.2580D 03	-0.6600D 03	0.7896D 03	0.9108D 03				
0.2000D 03	-0.4480D 03	0.5520D 03	0.2560D 03	0.4000D 02	0.0	-0.1040D 03	0.6640D 03				
0.1600D 02	-0.3120D 03	-0.1040D 03	0.4080D 03	0.8800D 02	-0.1200D 03	-0.1680D 03	-0.2000D 02				
0.3536D 03	-0.6400D 01	0.4400D 03	0.2680D 03	0.4240D 02	-0.6880D 02	0.5048D 03	0.1184D 03				
0.1520D 04	0.8640D 03	-0.6102D 03	0.1228D 04	-0.1501D 04	-0.9450D 03	0.2403D 04	0.1296D 04				
-0.1809D 04	0.5670D 03	-0.1350D 03	-0.2163D 03	0.1269D 04	0.4050D 03	0.7830D 03	-0.5940D 03				
0.1323D 04	-0.7830D 03	0.1728D 04	0.2052D 04	0.1536D 04	-0.1021D 04	0.1031D 04	-0.5265D 03				

ELAPSED TIME = 10.91

ELAPSED TIME = 0.90

NUMBER OF ITERACTIONS = 0.800000000D 01

```
 OUTPUT NO.4, MINIMIZES V + BETA * U SQUARE ERRORS
DESIRED MATRIX IS
  MATRIX NUMBER       17,    6 ROWS, 6 COLUMNS
 0.9000D 01 -0.1388D-15 -0.7355D-15 -0.8327D-16 -0.3886D-15 -0.2665D-14
-0.1388D-15  0.1800D 02  0.3789D-14  0.1346D-14 -0.9437D-15 -0.1088D-13
-0.7355D-15  0.3789D-14  0.5000D 01 -0.1069D-14  0.3053D-15  0.4885D-14
-0.8327D-16  0.1346D-14 -0.1069D-14  0.1200D 02  0.1998D-14  0.8882D-15
-0.3886D-15 -0.9437D-15  0.3053D-15  0.1998D-14  0.8000D 01  0.1044D-13
-0.2665D-14 -0.1088D-13  0.4885D-14  0.8882D-15  0.1044D-13  0.2700D 02
 BETA =0.8000000D 00
 CRITERION NUMBER     1 EQUALS    1.797180D-22
 CRITERION NUMBER     2 EQUALS    3.555419D-25
 CRITERION NUMBER     3 EQUALS    4.837873D-24
 CRITERION NUMBER     4 EQUALS    1.000014D-22
 ELAPSED TIME =          0.99
 TOTAL ELAPSED TIME =         12.86
```

MATRIX P FOLLOWS

MATRIX NUMBER 1, 6 ROWS, 6 COLUMNS

0.9000D 01	0.0	0.0	0.0	0.0	0.0
0.0	0.1800D 02	0.0	0.0	0.0	0.0
0.0	0.0	0.5000D 01	0.0	0.0	0.0
0.0	0.0	0.0	0.1200D 02	0.0	0.0
0.0	0.0	0.0	0.0	0.8000D 01	0.0
0.0	0.0	0.0	0.0	0.0	0.2700D 02

MATRIX BE FOLLOWS

MATRIX NUMBER 2, 6 ROWS,24 COLUMNS

-0.9700D 02	0.9200D 02	-0.1900D 02	0.3800D 02	-0.9900D 02	0.2430D 02	0.7000D 00	-0.5020D 02
0.3200D 02	-0.8840D 02	0.0	0.5100D 02	0.7500D 02	0.1200D 02	-0.2800D 02	0.6240D 02
-0.5000D 00	0.5160D 02	-0.7160D 02	0.4300D 01	0.1460D 02	0.2350D 02	-0.3670D 02	0.4500D 02
0.9000D 01	0.4500D 02	-0.2000D 02	0.8700D 02	0.9700D 02	0.1050D 02	0.2080D 02	-0.5210D 02
0.9870D 02	-0.2690D 02	0.9500D 02	0.1300D 02	-0.7900D 02	0.6700D 02	0.1700D 02	0.2500D 01
-0.5470D 02	0.3590D 02	-0.6550D 02	0.5550D 02	-0.5000D 00	0.7510D 02	0.3450D 02	0.8210D 02
0.4200D 02	-0.2700D 02	0.3400D 02	0.1900D 02	0.6200D 02	0.3500D 02	0.0	0.8700D 02
-0.5100D 02	0.3500D 02	0.1100D 02	-0.5300D 02	0.2400D 02	0.6000D 02	0.1500D 02	-0.3200D 02
-0.6900D 02	0.4200D 02	0.3100D 02	0.1500D 02	0.9810D 02	-0.4600D 01	0.3200D 01	0.6630D 02
0.1400D 02	0.3400D 02	-0.1500D 02	0.0	-0.8500D 02	0.5100D 02	-0.3900D 02	0.7800D 02
0.6400D 02	0.4100D 02	0.4100D 02	-0.2100D 02	-0.2300D 02	0.3600D 02	0.6100D 02	0.3100D 02
-0.1600D 02	-0.2000D 01	0.6700D 02	-0.5800D 02	0.2150D 02	-0.5500D 02	0.6580D 02	0.7590D 02
0.2500D 02	-0.5600D 02	0.6900D 02	0.3200D 02	0.5000D 01	0.0	-0.1300D 02	0.8300D 02
0.2000D 01	-0.3900D 02	-0.1300D 02	0.5100D 02	0.1100D 02	-0.1500D 02	-0.2100D 02	-0.2500D 01
0.4420D 02	-0.8000D 00	0.5500D 02	0.3350D 02	0.5300D 01	-0.8600D 01	0.6310D 02	0.1480D 02
0.5630D 02	0.3200D 02	-0.2260D 02	0.4550D 02	-0.5560D 02	-0.3500D 02	0.8900D 02	0.4800D 02
-0.6700D 02	0.2100D 02	-0.5000D 01	-0.8000D 01	0.4700D 02	0.1500D 02	0.2900D 02	-0.2200D 02
0.4900D 02	-0.2900D 02	0.6400D 02	0.7600D 02	0.5690D 02	-0.3780D 02	0.3820D 02	-0.1950D 02

MATRIX EB FOLLOWS

MATRIX NUMBER 3, 6 ROWS, 24 COLUMNS

```
-0.873CD 03  0.8280D 03 -0.1710D 03  0.3420D 03 -0.8910D 03  0.2187D 03  0.6300D 01 -0.4518D 03
 0.2880D 03 -0.7956D 03  0.0          0.4590D 03  0.6750D 03  0.1080D 03 -0.2520D 03  0.5616D 03
-0.450CD 01  0.4644D 03 -0.6444D 03  0.3870D 02  0.1314D 03  0.2115D 03 -0.3303D 03  0.4050D 03
 0.1620D 03  0.8100D 03 -0.3600D 03  0.1566D 04  0.1746D 04  0.1890D 03  0.3744D 03 -0.9378D 03
 0.1777D 04 -0.4842D 03  0.1710D 04  0.2340D 03 -0.1422D 04  0.1206D 04  0.3060D 03  0.4500D 02
-0.9846D 03  0.6462D 03 -0.1179D 04  0.9990D 03 -0.9000D 01  0.1352D 04  0.6210D 03  0.1478D 04
 0.2100D 03 -0.1350D 03  0.1700D 03  0.9500D 02  0.3100D 03  0.1750D 03  0.0          0.4350D 03
-0.2550D 03  0.1750D 03  0.5500D 02 -0.2650D 03  0.1200D 03  0.3000D 03  0.7500D 02 -0.1600D 03
-0.3450D 03  0.2100D 03  0.1550D 03  0.7500D 02  0.4905D 03 -0.2300D 02  0.1600D 02  0.3315D 03
 0.1680D 03  0.4080D 03 -0.1800D 03  0.0         -0.1020D 04  0.6120D 03 -0.4680D 03  0.9360D 03
 0.7680D 03  0.4920D 03  0.4920D 03 -0.2520D 03 -0.2760D 03  0.4320D 03  0.7320D 03  0.3720D 03
-0.1920D 03 -0.2400D 02  0.8040D 03 -0.6960D 03  0.2580D 03 -0.6600D 03  0.7896D 03  0.9108D 03
 0.2000D 03 -0.4480D 03  0.5520D 03  0.2560D 03  0.4000D 02  0.0         -0.1040D 03  0.6640D 03
 0.1600D 02 -0.3120D 03 -0.1040D 03  0.4080D 03  0.8800D 02 -0.1200D 03 -0.1680D 03 -0.2000D 02
 0.3536D 03 -0.6400D 01  0.4400D 03  0.2680D 03  0.4240D 02 -0.6880D 02  0.5048D 03  0.1184D 03
 0.1520D 04  0.8640D 03 -0.6102D 03  0.1228D 04 -0.1501D 04 -0.9450D 03  0.2403D 04  0.1296D 04
-0.1809D 04  0.5670D 03 -0.1350D 03 -0.2160D 03  0.1269D 04  0.4050D 03  0.7830D 03 -0.5940D 03
 0.1323D 04 -0.7830D 03  0.1728D 04  0.2052D 04  0.1536D 04 -0.1021D 04  0.1031D 04 -0.5265D 03
```

ERROR IN BLANCA = 0.01000

ERROR IN BLANCA = 0.01000

MATRIX U FOLLOWS

MATRIX NUMBER 4, 6 ROWS, 24 COLUMNS

```
-0.1052D 03  0.8658D 02 -0.2068D 02  0.4062D 02 -0.9314D 02  0.2531D 02  0.7619D 00 -0.5323D 02
 0.2964D 02 -0.9087D 02  0.2318D 00  0.4927D 02  0.6915D 02  0.1111D 02 -0.2937D 02  0.5892D 02
-0.5106D 00  0.5199D 02 -0.7779D 02  0.3884D 01  0.1384D 02  0.2411D 02 -0.3850D 02  0.4176D 02
 0.8827D 01  0.4064D 02 -0.2122D 02  0.9073D 02  0.8744D 02  0.1096D 02  0.2265D 02 -0.5227D 02
 0.9674D 02 -0.2810D 02  0.1029D 03  0.1231D 02 -0.8549D 02  0.7140D 02  0.1641D 02  0.2675D 01
-0.5910D 02  0.3826D 02 -0.6028D 02  0.5143D 02 -0.4528D 00  0.7832D 02  0.3386D 02  0.7743D 02
 0.3815D 02 -0.2780D 02  0.3433D 02  0.1980D 02  0.6712D 02  0.3157D 02  0.2069D 00  0.9459D 02
-0.5325D 02  0.3524D 02  0.9924D 01 -0.5662D 02  0.2505D 02  0.6025D 02  0.1386D 02 -0.3158D 02
-0.6931D 02  0.4202D 02  0.3309D 02  0.1365D 02  0.9464D 02 -0.4179D 01  0.2968D 01  0.6291D 02
 0.1522D 02  0.3199D 02 -0.1602D 02  0.5648D-01 -0.9211D 02  0.5216D 02 -0.4008D 02  0.7667D 02
 0.6430D 02  0.4127D 02  0.3946D 02 -0.1934D 02 -0.2357D 02  0.3455D 02  0.6634D 02  0.3124D 02
-0.1679D 02 -0.2094D 01  0.6757D 02 -0.6069D 02  0.1946D 02 -0.4956D 02  0.7113D 02  0.7766D 02
 0.2273D 02 -0.5197D 02  0.6779D 02  0.3337D 02  0.5115D 01  0.4043D 00 -0.1226D 02  0.8964D 02
 0.1853D 01 -0.3511D 02 -0.1389D 02  0.4827D 02  0.1037D 02 -0.1360D 02 -0.2278D 02 -0.2450D 01
 0.4549D 02 -0.8394D 00  0.5651D 02  0.3226D 02  0.5741D 01 -0.9329D 01  0.6715D 02  0.1427D 02
 0.5346D 02  0.3151D 02 -0.2095D 02  0.4126D 02 -0.5709D 02 -0.3303D 02  0.8546D 02  0.4351D 02
-0.6054D 02  0.2076D 02 -0.4913D 01 -0.8333D 01  0.4795D 02  0.1575D 02  0.2921D 02 -0.2251D 02
 0.4815D 02 -0.2964D 02  0.6181D 02  0.7452D 02  0.6197D 02 -0.4012D 02  0.3980D 02 -0.2071D 02
```

MATRIX V FOLLOWS

MATRIX NUMBER 5, 6 ROWS,24 COLUMNS

-0.9166D 03	0.8613D 03	-0.1583D 03	0.3649D 03	-0.9509D 03	0.2331D 03	0.6178D 01	-0.4439D 03
0.2950D 03	-0.8119D 03	0.8306D 00	0.4871D 03	0.6817D 03	0.1047D 03	-0.2535D 03	0.5768D 03
-0.4334D 01	0.4954D 03	-0.6116D 03	0.4212D 02	0.1372D 03	0.2082D 03	-0.3194D 03	0.3846D 03
0.1599D 03	0.7295D 03	-0.3537D 03	0.1674D 04	0.1830D 04	0.1716D 03	0.3467D 03	-0.9999D 03
0.1850D 04	-0.4361D 03	0.1644D 04	0.2109D 03	-0.1337D 04	0.1117D 04	0.3021D 03	0.4347D 02
-0.1011D 04	0.6908D 03	-0.1109D 04	0.1085D 04	-0.8327D 01	0.1329D 04	0.5754D 03	0.1521D 04
0.2075D 03	-0.1254D 03	0.1608D 03	0.9320D 02	0.3242D 03	0.1728D 03	-0.9025D 00	0.4184D 03
-0.2565D 03	0.1902D 03	0.4969D 02	-0.2491D 03	0.1205D 03	0.2769D 03	0.8139D 02	-0.1732D 03
-0.3318D 03	0.1937D 03	0.1681D 03	0.7065D 02	0.4526D 03	-0.2274D 02	0.1606D 02	0.3519D 03
0.1586D 03	0.4164D 03	-0.1665D 03	0.8025D 00	-0.1050D 04	0.5510D 03	-0.4714D 03	0.9405D 03
0.7717D 03	0.5217D 03	0.4755D 03	-0.2292D 03	-0.2835D 03	0.4693D 03	0.6909D 03	0.3500D 03
-0.1965D 03	-0.2517D 02	0.7592D 03	-0.7166D 03	0.2739D 03	-0.6518D 03	0.7181D 03	0.9916D 03
0.2135D 03	-0.4431D 03	0.4986D 03	0.2510D 03	0.4003D 02	0.4526D 00	-0.1042D 03	0.6023D 03
0.1732D 02	-0.3389D 03	-0.1068D 03	0.3769D 03	0.8488D 02	-0.1280D 03	-0.1619D 03	-0.2171D 02
0.3215D 03	-0.6794D 01	0.3982D 03	0.2751D 03	0.4600D 02	-0.6473D 02	0.4675D 03	0.1290D 03
0.1554D 04	0.7781D 03	-0.5619D 03	0.1262D 04	-0.1424D 04	-0.1002D 04	0.2274D 04	0.1223D 04
-0.1857D 04	0.5720D 03	-0.1464D 03	-0.2086D 03	0.1147D 04	0.3728D 03	0.7404D 03	-0.6110D 03
0.1225D 04	-0.8190D 03	0.1702D 04	0.1861D 04	0.1672D 04	-0.1107D 04	0.9753D 03	-0.4759D 03

ELAPSED TIME = 10.95

ELAPSED TIME = 1.86

NUMBER OF ITERACTIONS = 0.17000000D 02

OUTPUT NO.4, MINIMIZES V + BETA * U SQUARE ERRORS

DESIRED MATRIX IS

MATRIX NUMBER 17, 6 ROWS, 6 COLUMNS

```
 0.9264D C1 -0.1389D CO -C.1716D-C1  0.3280D CO  0.1555D 00 -0.1794D 00
-0.1389D CO  0.1793D 02  0.1632D 00 -0.3352D CO  0.1642D 00 -0.1288D 00
-0.1716D-01  0.1632D CO  C.4651D C1  0.2899D 00 -0.9420D-01  0.3705D 00
 0.3280D CO -0.3352D 00  0.2899D 00  0.1170D 02 -0.2030D 00 -0.2331D 00
 0.1555D CO  0.1642D CO -0.9420D-C1 -0.2030D 00  0.7575D 01  0.3344D-01
-0.1794D 00 -0.1288D 00  C.3705D CO -0.2331D 00  0.3344D-01  0.2671D 02
```

BETA =0.8000C0D 00

CRITERION NUMBER 1 EQUALS 4.998249D 05

CRITERION NUMBER 2 EQUALS 2.57C142D 03

CRITERION NUMBER 3 EQUALS 3.139593D 04

CRITERION NUMBER 4 EQUALS 2.7882280 05

ELAPSED TIME = 0.99

TOTAL ELAPSED TIME = 13.85

APPENDIX D

BRIEF TREATMENT OF RELATED PROBLEMS

D-1 Case where there are no Restrictions on C.

In the development of this thesis, starting with the matrix equation CU = V, matrix C was restricted to be symmetric and positive definite. Suppose now that these restrictions should be removed, and, indeed, let it only be required that C have the correct dimensions to relate matrices U and V which may have a different number of rows.

Let us write equation (4) in a modified form

$$\alpha = tr\left[(CU-V)^T Z (CU-V)\right] \qquad \text{(D-1)}$$

where Z is arbitrary and might be thought of as representing a set of weighting factors. If Z is the unit matrix, equation (D-1) reduces to equation (4).

From equation (D-1)

$$\alpha = tr\left[U^T C^T Z C U - U^T C^T Z V - V^T Z C U + V^T Z V\right]$$

Note that, in general, $C^T \neq C$ since the requirement of symmetry is not made here.

Using * to denote partial differentiation with respect to any element x, as before, one gets

$$\alpha^* = tr\left[U^T C^{T*} Z C U + U^T C^T Z C^* U - U^T C^{T*} Z V - V^T Z C^* U\right]$$

$$= tr\left[\left(C U U^T - V U^T\right)\left(C^{T*}\right)\left(Z + Z^T\right)\right] \qquad \text{(D-2)}$$

Thus α is made stationary with respect to each element of C if C satisfies the equation

$$CUU^T = VU^T$$

If U is of rank m, where m is the number of rows in U, then

$$C = VU^T(UU^T)^{-1}$$

It is interesting to note, that the result is independent of the arbitrary weighting matrix Z.

D-2 Case where C is Orthogonal.

This problem arises in the treatment of the orientation of a rigid body.

If C is orthogonal, then

$$C^T = C^{-1} = Y$$

where the notation $Y = C^{-1}$ is used for convenience in manipulation.

As before let x be any element of C, and let * denote $\frac{\partial}{\partial x}$.
Then the relation CY = I, leads to C* = -CY*C, and equation (4) can be written as

$$\alpha = tr\left(U^TU - U^TYV - V^TCU + V^TV\right)$$

so that

$$\alpha^* = tr\left(-U^TY^*V + V^TCY^*CU\right)$$

$$= tr\left[\left(Y^*\right)\left(CUV^TC - VU^T\right)\right] \qquad \text{(D-3)}$$

Thus, α is stationary with respect to variation in any of the elements of C, if C satisfies the equation

$$CUV^TC = VU^T$$

which will be so if C is any of the several values

$$C = (VU^TUV^T)^{\frac{1}{2}}(UV^T)^{-1}$$

VU^TUV^T is symmetric matrix, and its square root can be obtained as described in Section 4.

These results were obtained by Professor Brock and are given here so as to collect in one place information on this general matrix problem.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, Calif. 93940	2
3.	Professor John E. Brock Mechanical Engineering Department Naval Postgraduate School Monterey, Calif. 93940	3
4.	Department of Mechanical Engineering Naval Postgraduate School Monterey, Calif. 93940	1
5.	Mr. Carl P. Wilson Code 452 General Dynamics/Electric Boat Division Eastern Point Road Groton, Conn. 03640	3
6.	Mr. Leslie Kaldor Navy Marine Engineering Laboratory Annapolis, Maryland 21402	3
7.	LT Cesar F. Villaran Tapia, Peruvian Navy Moreyra y Riglos 769, Pueblo Libre Lima, Peru. S.A.	4
8.	Commander Naval Ship Systems Command Navy Department Washington, D. C. 20360	1

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Naval Postgraduate School Monterey, California 93940	Unclassified
	2b. GROUP

3. REPORT TITLE

Determination of Optimal Compliance and Stiffness Matrices from Experimental Data

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

None

5. AUTHOR(S) *(Last name, first name, initial)*

VILLAMAR TAPIA, Cesar E., LT., Peruvian Navy

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
September 1967	102	7

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO. N/A	N/A
c. N/A	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	N/A

10. AVAILABILITY/LIMITATION NOTICES

[redacted] to foreign nationals may be made only with prior approval of the Naval Postgraduate School.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
N/A	Naval Postgraduate School

13. ABSTRACT

A linear structure can be characterized by its compliance matrix C, which is 6x6 symmetrical and positive definite and which relates a force 6-vector $\bar{F}_j$ to a displacement 6-vector $\bar{D}_j$ by the relation $C\bar{F}_j = \bar{D}_j$. The inverse, S, of C, is called the stiffness matrix and satisfies $S\bar{D}_j = \bar{F}_j$. This thesis deals with the problem of finding optimal values of such matrices C and S from experimental determinations of a sufficient number of vector-pairs $(\bar{F}_j, \bar{D}_j)$ which are presumed to contain random errors.

J. E. Brock has introduced this problem area, suggested several different criteria of optimality, and solved some of the corresponding specific problems. This thesis completes the solution to a previously unsolved specific problem of this group and contributes computationally convenient new solutions to another. Moreover, a computer program, originally written for the CDC 1604 has been rewritten, in FORTRAN IV Language, as two programs for the IBM System 360 computer, and the capability has been significantly augmented.

DD FORM 1 JAN 64 1473

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Matrices						
Compliance						
Stiffness						
Analysis of Data						
Experimental Data						
Data Reduction						
Linear Systems						
Least-Squares						
Experiment						
Optimal Matrices						
Matrix Equations						
Matrix Square Roots